



Pipelines



Pipeline [tekton.dev/v1]

Description

Pipeline describes a list of Tasks to execute. It expresses how outputs of tasks feed into inputs of subsequent tasks.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
<code>metadata</code>	<code>ObjectMeta</code>	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the desired state of the Pipeline from the client

.spec

Description

Spec holds the desired state of the Pipeline from the client

Type

`object`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a user-facing description of the pipeline that may be used to populate a UI.

Property	Type	Description
<code>displayName</code>	<code>string</code>	DisplayName is a user-facing name of the pipeline that may be used to populate a UI.
<code>finally</code>	<code>array</code>	Finally declares the list of Tasks that execute just before leaving the Pipeline i.e. either after all Tasks are finished executing successfully or after a failure which would result in ending the Pipeline
<code>params</code>	<code>array</code>	Params declares a list of input parameters that must be supplied when this Pipeline is run.
<code>results</code>	<code>array</code>	Results are values that this pipeline can output once run
<code>tasks</code>	<code>array</code>	Tasks declares the graph of Tasks that execute when this Pipeline is run.
<code>workspaces</code>	<code>array</code>	Workspaces declares a set of named workspaces that are expected to be provided by a PipelineRun.

`.spec.finally`

Description

Finally declares the list of Tasks that execute just before leaving the Pipeline i.e. either after all Tasks are finished executing successfully or after a failure which would result in ending the Pipeline

Type

array

.spec.finally[]

Description

PipelineTask defines a task in a Pipeline, passing inputs from both Params and from the output of previous tasks.

Type

object

Property	Type	Description
<code>description</code>	<code>string</code>	Description is the description of this task within the context of a Pipeline. This description may be used to populate a UI.
<code>displayName</code>	<code>string</code>	DisplayName is the display name of this task within the context of a Pipeline. This display name may be used to populate a UI.
<code>matrix</code>	<code>object</code>	Matrix declares parameters used to fan out this task.
<code>name</code>	<code>string</code>	Name is the name of this task within the context of a Pipeline. Name is used as a coordinate with the <code>from</code> and <code>runAfter</code> fields to establish the execution order of tasks relative to one another.

Property	Type	Description
<code>onError</code>	<code>string</code>	OnError defines the exiting behavior of a PipelineRun on error can be set to [continue stopAndFail]
<code>params</code>	<code>array</code>	Parameters declares parameters passed to this task.
<code>pipelineRef</code>	<code>object</code>	PipelineRef is a reference to a pipeline definition Note: PipelineRef is in preview mode and not yet supported
<code>pipelineSpec</code>	<code></code>	PipelineSpec is a specification of a pipeline Note: PipelineSpec is in preview mode and not yet supported Specifying PipelineSpec can be disabled by setting <code>disable-inline-spec</code> feature flag. See Pipeline.spec (API version: tekton.dev/v1)
<code>retries</code>	<code>integer</code>	Retries represents how many times this task should be retried in case of task failure: ConditionSucceeded set to False
<code>runAfter</code>	<code>array</code>	RunAfter is the list of PipelineTask names that should be executed before this Task executes. (Used to force a specific ordering in graph execution.)
<code>taskRef</code>	<code>object</code>	TaskRef is a reference to a task definition.

Property	Type	Description
<code>taskSpec</code>	<code>object</code>	TaskSpec is a specification of a task Specifying TaskSpec can be disabled by setting <code>disable-inline-spec</code> feature flag. See Task.spec (API version: tekton.dev/v1)
<code>timeout</code>	<code>string</code>	Time after which the TaskRun times out. Defaults to 1 hour. Refer Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration ↗
<code>when</code>	<code>array</code>	When is a list of when expressions that need to be true for the task to run
<code>workspaces</code>	<code>array</code>	Workspaces maps workspaces from the pipeline spec to the workspaces declared in the Task.

`.spec.finally[].matrix`

Description

Matrix declares parameters used to fan out this task.

Type

`object`

Property	Type	Description
<code>include</code>	<code>array</code>	Include is a list of IncludeParams which allows passing in specific combinations of Parameters into the Matrix.

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of parameters used to fan out the pipelineTask Params takes only <code>Parameters</code> of type <code>"array"</code> Each array element is supplied to the <code>PipelineTask</code> by substituting <code>params</code> of type <code>"string"</code> in the underlying <code>Task</code> . The names of the <code>params</code> in the <code>Matrix</code> must match the names of the <code>params</code> in the underlying <code>Task</code> that they will be substituting.

`.spec.finally[].matrix.include`

Description

Include is a list of IncludeParams which allows passing in specific combinations of Parameters into the Matrix.

Type

`array`

`.spec.finally[].matrix.include[]`

Description

IncludeParams allows passing in a specific combinations of Parameters into the Matrix.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name the specified combination
<code>params</code>	<code>array</code>	Params takes only <code>Parameters</code> of type <code>"string"</code> The names of the <code>params</code> must match the names of the <code>params</code> in the

Property	Type	Description
		underlying <code>Task</code>

`.spec.finally[].matrix.include[].params`

Description

Params takes only `Parameters` of type `string` The names of the `params` must match the names of the `params` in the underlying `Task`

Type

array

`.spec.finally[].matrix.include[].params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

`.spec.finally[].matrix.params`

Description

Params is a list of parameters used to fan out the pipelineTask Params takes only `Parameters` of type `array` Each array element is supplied to the `PipelineTask` by

substituting `params`` of type `"string"` in the underlying `Task``. The names of the `params`` in the `Matrix`` must match the names of the `params`` in the underlying `Task`` that they will be substituting.

Type

array

`.spec.finally[].matrix.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

`.spec.finally[].params`

Description

Parameters declares parameters passed to this task.

Type

array

`.spec.finally[].params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

.spec.finally[].pipelineRef

Description

PipelineRef is a reference to a pipeline definition Note: PipelineRef is in preview mode and not yet supported

Type

`object`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	API version of the referent
<code>name</code>	<code>string</code>	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names ↗
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Property	Type	Description
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.finally[].pipelineRef.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.finally[].pipelineRef.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

`.spec.finally[].runAfter`

Description

RunAfter is the list of PipelineTask names that should be executed before this Task executes. (Used to force a specific ordering in graph execution.)

Type

array

.spec.finally[].runAfter[]

Type

string

.spec.finally[].taskRef

Description

TaskRef is a reference to a task definition.

Type

object

Property	Type	Description
apiVersion	string	API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task
kind	string	TaskKind indicates the Kind of the Task: - Namespaced Task when Kind is set to "Task". If Kind is "", it defaults to "Task". - Custom Task when Kind is non-empty and APIVersion is non-empty

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names ↗
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.finally[].taskRef.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.finally[].taskRef.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

name

value

Property	Type	Description
name	string	
value		

.spec.finally[].when

Description

When is a list of when expressions that need to be true for the task to run

Type

array

.spec.finally[].when[]

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

object

Property	Type	Description
cel	string	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md ↗

Property	Type	Description
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.spec.finally[].when[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

`array`

`.spec.finally[].when[].values[]`

Type

`string`

`.spec.finally[].workspaces`

Description

Workspaces maps workspaces from the pipeline spec to the workspaces declared in the Task.

Type

array

.spec.finally[].workspaces[]

Description

WorkspacePipelineTaskBinding describes how a workspace passed into the pipeline should be mapped to a task's declared workspace.

Type

object

Required

name

Property	Type	Description
name	string	Name is the name of the workspace as declared by the task
subPath	string	SubPath is optionally a directory on the volume which should be used for this binding (i.e. the volume will be mounted at this sub directory).
workspace	string	Workspace is the name of the workspace declared by the pipeline

.spec.params

Description

Params declares a list of input parameters that must be supplied when this Pipeline is run.

Type

array

.spec.params[]

Description

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources). Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

Type

object

Required

name

Property	Type	Description
<code>default</code>		Default is the value a parameter takes if no input value is supplied. If default is set, a Task may be executed without a supplied value for the parameter.
<code>description</code>	<code>string</code>	Description is a user-facing description of the parameter that may be used to populate a UI.
<code>enum</code>	<code>array</code>	Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.
<code>name</code>	<code>string</code>	Name declares the name by which a parameter is referenced.
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs parameter.

Property	Type	Description
<code>type</code>	<code>string</code>	Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

`.spec.params[].enum`

Description

Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Type

`array`

`.spec.params[].enum[]`

Type

`string`

`.spec.params[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs parameter.

Type

`object`

`.spec.results`

Description

Results are values that this pipeline can output once run

Type

`array`

`.spec.results[]`

Description

PipelineResult used to describe the results of a pipeline

Type

`object`

Required

`name``value`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a human-readable description of the result
<code>name</code>	<code>string</code>	Name the given name
<code>type</code>	<code>string</code>	Type is the user-specified type of the result. The possible types are 'string', 'array', and 'object', with 'string' as the default. 'array' and 'object' types are alpha features.
<code>value</code>		Value the expression used to retrieve the value

`.spec.tasks`

Description

Tasks declares the graph of Tasks that execute when this Pipeline is run.

Type

`array`

`.spec.tasks[]`

Description

PipelineTask defines a task in a Pipeline, passing inputs from both Params and from the output of previous tasks.

Type

`object`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is the description of this task within the context of a Pipeline. This description may be used to populate a UI.
<code>displayName</code>	<code>string</code>	DisplayName is the display name of this task within the context of a Pipeline. This display name may be used to populate a UI.
<code>matrix</code>	<code>object</code>	Matrix declares parameters used to fan out this task.
<code>name</code>	<code>string</code>	Name is the name of this task within the context of a Pipeline. Name is used as a coordinate with the <code>from</code> and <code>runAfter</code> fields to establish the execution order of tasks relative to one another.

Property	Type	Description
<code>onError</code>	<code>string</code>	OnError defines the exiting behavior of a PipelineRun on error can be set to [continue stopAndFail]
<code>params</code>	<code>array</code>	Parameters declares parameters passed to this task.
<code>pipelineRef</code>	<code>object</code>	PipelineRef is a reference to a pipeline definition Note: PipelineRef is in preview mode and not yet supported
<code>pipelineSpec</code>	<code></code>	PipelineSpec is a specification of a pipeline Note: PipelineSpec is in preview mode and not yet supported Specifying PipelineSpec can be disabled by setting <code>disable-inline-spec</code> feature flag. See Pipeline.spec (API version: tekton.dev/v1)
<code>retries</code>	<code>integer</code>	Retries represents how many times this task should be retried in case of task failure: ConditionSucceeded set to False
<code>runAfter</code>	<code>array</code>	RunAfter is the list of PipelineTask names that should be executed before this Task executes. (Used to force a specific ordering in graph execution.)
<code>taskRef</code>	<code>object</code>	TaskRef is a reference to a task definition.

Property	Type	Description
<code>taskSpec</code>	<code>object</code>	TaskSpec is a specification of a task Specifying TaskSpec can be disabled by setting <code>disable-inline-spec</code> feature flag. See Task.spec (API version: tekton.dev/v1)
<code>timeout</code>	<code>string</code>	Time after which the TaskRun times out. Defaults to 1 hour. Refer Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration ↗
<code>when</code>	<code>array</code>	When is a list of when expressions that need to be true for the task to run
<code>workspaces</code>	<code>array</code>	Workspaces maps workspaces from the pipeline spec to the workspaces declared in the Task.

`.spec.tasks[].matrix`

Description

Matrix declares parameters used to fan out this task.

Type

`object`

Property	Type	Description
<code>include</code>	<code>array</code>	Include is a list of IncludeParams which allows passing in specific combinations of Parameters into the Matrix.

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of parameters used to fan out the pipelineTask Params takes only <code>Parameters</code> of type <code>"array"</code> Each array element is supplied to the <code>PipelineTask</code> by substituting <code>params</code> of type <code>"string"</code> in the underlying <code>Task</code> . The names of the <code>params</code> in the <code>Matrix</code> must match the names of the <code>params</code> in the underlying <code>Task</code> that they will be substituting.

`.spec.tasks[].matrix.include`

Description

Include is a list of IncludeParams which allows passing in specific combinations of Parameters into the Matrix.

Type

`array`

`.spec.tasks[].matrix.include[]`

Description

IncludeParams allows passing in a specific combinations of Parameters into the Matrix.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name the specified combination
<code>params</code>	<code>array</code>	Params takes only <code>Parameters</code> of type <code>"string"</code> The names of the <code>params</code> must match the names of the <code>params</code> in the

Property	Type	Description
		underlying <code>Task</code>

`.spec.tasks[].matrix.include[].params`

Description

Params takes only `Parameters` of type `string` The names of the `params` must match the names of the `params` in the underlying `Task`

Type

array

`.spec.tasks[].matrix.include[].params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

`.spec.tasks[].matrix.params`

Description

Params is a list of parameters used to fan out the pipelineTask Params takes only `Parameters` of type `array` Each array element is supplied to the `PipelineTask` by

substituting `params` of type `"string"` in the underlying `Task`. The names of the `params` in the `Matrix` must match the names of the `params` in the underlying `Task` that they will be substituting.

Type

array

`.spec.tasks[].matrix.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

`.spec.tasks[].params`

Description

Parameters declares parameters passed to this task.

Type

array

`.spec.tasks[].params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

.spec.tasks[].pipelineRef

Description

PipelineRef is a reference to a pipeline definition Note: PipelineRef is in preview mode and not yet supported

Type

object

Property	Type	Description
apiVersion	string	API version of the referent
name	string	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names
params	array	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Property	Type	Description
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.tasks[].pipelineRef.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.tasks[].pipelineRef.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

`.spec.tasks[].runAfter`

Description

RunAfter is the list of PipelineTask names that should be executed before this Task executes. (Used to force a specific ordering in graph execution.)

Type

array

.spec.tasks[].runAfter[]

Type

string

.spec.tasks[].taskRef

Description

TaskRef is a reference to a task definition.

Type

object

Property	Type	Description
apiVersion	string	API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task
kind	string	TaskKind indicates the Kind of the Task: - Namespaced Task when Kind is set to "Task". If Kind is "", it defaults to "Task". - Custom Task when Kind is non-empty and APIVersion is non-empty

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names ↗
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.tasks[].taskRef.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.tasks[].taskRef.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

name

value

Property	Type	Description
name	string	
value		

.spec.tasks[].when

Description

When is a list of when expressions that need to be true for the task to run

Type

array

.spec.tasks[].when[]

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

object

Property	Type	Description
cel	string	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md

Property	Type	Description
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.spec.tasks[].when[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

`array`

`.spec.tasks[].when[].values[]`

Type

`string`

`.spec.tasks[].workspaces`

Description

Workspaces maps workspaces from the pipeline spec to the workspaces declared in the Task.

Type

array

`.spec.tasks[].workspaces[]`

Description

WorkspacePipelineTaskBinding describes how a workspace passed into the pipeline should be mapped to a task's declared workspace.

Type

object

Required

name

Property	Type	Description
name	string	Name is the name of the workspace as declared by the task
subPath	string	SubPath is optionally a directory on the volume which should be used for this binding (i.e. the volume will be mounted at this sub directory).
workspace	string	Workspace is the name of the workspace declared by the pipeline

`.spec.workspaces`

Description

Workspaces declares a set of named workspaces that are expected to be provided by a PipelineRun.

Type

`array`

`.spec.workspaces[]`

Description

PipelineWorkspaceDeclaration creates a named slot in a Pipeline that a PipelineRun is expected to populate with a workspace binding.

Type

`object`

Required

`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a human readable string describing how the workspace will be used in the Pipeline. It can be useful to include a bit of detail about which tasks are intended to have access to the data on the workspace.
<code>name</code>	<code>string</code>	Name is the name of a workspace to be provided by a PipelineRun.
<code>optional</code>	<code>boolean</code>	Optional marks a Workspace as not being required in PipelineRuns. By default this field is false and so declared workspaces are required.

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1/namespaces/{namespace}/pipelines`
 - **DELETE** : delete collection of Pipeline
 - **GET** : list objects of kind Pipeline
 - **POST** : create a new Pipeline
- `/apis/tekton.dev/v1/namespaces/{namespace}/pipelines/{name}`
 - **DELETE** : delete the specified Pipeline
 - **GET** : read the specified Pipeline
 - **PATCH** : partially update the specified Pipeline
 - **PUT** : replace the specified Pipeline

`/apis/tekton.dev/v1/namespaces/{namespace}/pipelines`

HTTP method

DELETE

Description

delete collection of Pipeline

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Pipeline

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineList</code> schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new Pipeline

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Pipeline</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>Pipeline</code> schema
201 - Created	<code>Pipeline</code> schema
202 - Accepted	<code>Pipeline</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1/namespaces/{namespace}/pipelines/{name}

HTTP method

DELETE

Description

delete the specified Pipeline

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
202 - Accepted	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Pipeline

HTTP responses

HTTP code	Response body
200 - OK	Pipeline schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Pipeline

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	string	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>Pipeline</code> schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified Pipeline

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+. - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Pipeline</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>Pipeline</code> schema
201 - Created	<code>Pipeline</code> schema
401 - Unauthorized	Empty



Task [tekton.dev/v1]

Description

Task represents a collection of sequential steps that are run as part of a Pipeline using a set of inputs and producing a set of outputs. Tasks execute when TaskRuns are created that provide the input parameters and resources and output resources the Task requires.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

Property	Type	Description
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
<code>metadata</code>	<code>ObjectMeta</code>	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the desired state of the Task from the client

.spec

Description

Spec holds the desired state of the Task from the client

Type

`object`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a user-facing description of the task that may be used to populate a UI.
<code>displayName</code>	<code>string</code>	DisplayName is a user-facing name of the task that may be used to populate a UI.

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of input parameters required to run the task. Params must be supplied as inputs in TaskRuns unless they declare a default value.
<code>results</code>	<code>array</code>	Results are values that this Task can output
<code>sidecars</code>	<code>array</code>	Sidecars are run alongside the Task's step containers. They begin before the steps start and end after the steps complete.
<code>stepTemplate</code>	<code>object</code>	StepTemplate can be used as the basis for all step containers within the Task, so that the steps inherit settings on the base container.
<code>steps</code>	<code>array</code>	Steps are the steps of the build; each step is run sequentially with the source mounted into /workspace.
<code>volumes</code>		Volumes is a collection of volumes that are available to mount into the steps of the build. See Pod.spec.volumes (API version: v1)
<code>workspaces</code>	<code>array</code>	Workspaces are the volumes that this Task requires.

`.spec.params`

Description

Params is a list of input parameters required to run the task. Params must be supplied as inputs in TaskRuns unless they declare a default value.

Type

array

.spec.params[]

Description

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources). Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

Type

object

Required

name

Property	Type	Description
<code>default</code>		Default is the value a parameter takes if no input value is supplied. If default is set, a Task may be executed without a supplied value for the parameter.
<code>description</code>	<code>string</code>	Description is a user-facing description of the parameter that may be used to populate a UI.
<code>enum</code>	<code>array</code>	Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Property	Type	Description
<code>name</code>	<code>string</code>	Name declares the name by which a parameter is referenced.
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs parameter.
<code>type</code>	<code>string</code>	Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

`.spec.params[].enum`

Description

Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Type

`array`

`.spec.params[].enum[]`

Type

`string`

`.spec.params[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs parameter.

Type

object

.spec.results

Description

Results are values that this Task can output

Type

array

.spec.results[]

Description

TaskResult used to describe the results of a task

Type

object

Required

name

Property	Type	Description
description	string	Description is a human-readable description of the result
name	string	Name the given name
properties	object	Properties is the JSON Schema properties to support key-value pairs results.

Property	Type	Description
<code>type</code>	<code>string</code>	Type is the user-specified type of the result. The possible type is currently "string" and will support "array" in following work.
<code>value</code>		Value the expression used to retrieve the value of the result from an underlying Step.

`.spec.results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type

`object`

`.spec.sidecars`

Description

Sidecars are run alongside the Task's step containers. They begin before the steps start and end after the steps complete.

Type

`array`

`.spec.sidecars[]`

Description

Sidecar has nearly the same data structure as Step but does not have the ability to timeout.

Type

`object`

Required

name

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>"\$(VAR_NAME)"</code> will produce the string literal <code>"\$(VAR_NAME)"</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>"\$(VAR_NAME)"</code> will produce the string literal <code>"\$(VAR_NAME)"</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell

Property	Type	Description
<code>computeResources</code>	<code>object</code>	ComputeResources required by this Sidecar. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/resources-containers/
<code>env</code>	<code>array</code>	List of environment variables to set in the Sidecar. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the Sidecar. The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the Sidecar is starting. When a key exists in multiple sources, the last associated with the last source will take precedence. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/envvars/#sources
<code>image</code>	<code>string</code>	Image reference name. More info: https://kubernetes.io/docs/concepts/containers/images/
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images/

Property	Type	Description
lifecycle	object	Actions that the management system should take Sidecar lifecycle events. Cannot be updated.
livenessProbe	object	Periodic probe of Sidecar liveness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes
name	string	Name of the Sidecar specified as a DNS_LABEL. in a Task must have a unique name (DNS_LABEL). Cannot be updated.
ports	array	List of ports to expose from the Sidecar. Exposing ports gives the system additional information about the connections a container uses, but is primarily informational. Specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default address inside a container will be accessible from the host. Cannot be updated.
readinessProbe	object	Periodic probe of Sidecar service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes

Property	Type	Description
<code>restartPolicy</code>	<code>string</code>	RestartPolicy refers to kubernetes RestartPolicy. It is set for an initContainer and must have it's policy set. It is currently left optional to help support Kubernetes prior to 1.29 when this feature was introduced.
<code>script</code>	<code>string</code>	Script is the contents of an executable file to execute. If Script is not empty, the Step cannot have an Command or Args.
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Sidecar should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>startupProbe</code>	<code>object</code>	StartupProbe indicates that the Pod the Sidecar is successfully initialized. If specified, no other probe is allowed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This is useful to provide different probe parameters at the beginning of the lifecycle, when it might take a long time to load data into the cache, than during steady-state operation. This can be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes ↗
<code>stdin</code>	<code>boolean</code>	Whether this Sidecar should allocate a buffer for stdin during container runtime. If this is not set, reads from stdin will fail.

Property	Type	Description
		Sidecar will always result in EOF. Default is false.
<code>stdinOnce</code>	<code>boolean</code>	Whether the container runtime should close the stdin stream after it has been opened by a single attach. When the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that opens stdin will keep stdin open and never receive an EOF. Default is false
<code>terminationMessagePath</code>	<code>string</code>	Optional: Path at which the file to which the Sidecar message will be written is mounted into the Sidecar container. Message written is intended to be brief final status of the container. Will be truncated by the container if it is greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.
<code>terminationMessagePolicy</code>	<code>string</code>	Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the Sidecar status message on both success and failure. FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 20 lines, whichever is smaller. Defaults to File. Cannot be updated.

Property	Type	Description
<code>tty</code>	<code>boolean</code>	Whether this Sidecar should allocate a TTY for itself. Requires 'stdin' to be true. Default is false.
<code>volumeDevices</code>	<code>array</code>	volumeDevices is the list of block devices to be used by the Sidecar.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Sidecar's filesystem. Cannot be updated.
<code>workingDir</code>	<code>string</code>	Sidecar's working directory. If not specified, the container runtime's default will be used, which might be configurable for the container image. Cannot be updated.
<code>workspaces</code>	<code>array</code>	This is an alpha field. You must set the "enable-alpha" flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Sidecar wants exclusive access to. Adding a workspace to the list means that any other Step or Sidecar that does not request this Workspace will not have access to it.

`.spec.sidecars[].args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are

reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.spec.sidecars[].args[]

Type

string

.spec.sidecars[].command

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.spec.sidecars[].command[]

Type

string

.spec.sidecars[].computeResources

Description

ComputeResources required by this Sidecar. Cannot be updated. More info:
<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
<code>claims</code>	<code>array</code>	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.sidecars[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.spec.sidecars[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.spec.sidecars[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.sidecars[].computeResources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.sidecars[].env`

Description

List of environment variables to set in the Sidecar. Cannot be updated.

Type

array

`.spec.sidecars[].env[]`

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.

Property	Type	Description
<code>value</code>	<code>string</code>	Variable references <code>\$(VAR_NAME)</code> are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to <code>""</code>.
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.sidecars[].env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code>, <code>metadata.namespace</code>, <code>metadata.labels['<KEY>']</code>, <code>metadata.annotations['<KEY>']</code>, <code>spec.nodeName</code>,

Property	Type	Description
		spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.sidecars[].env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.sidecars[].env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.sidecars[].env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.sidecars[].env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.sidecars[].envFrom`

Description

List of sources to populate environment variables in the Sidecar. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the container is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

`.spec.sidecars[].envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.spec.sidecars[].envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.spec.sidecars[].envFrom[].secretRef`

Description

The Secret to select from

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

`.spec.sidecars[].lifecycle`

Description

Actions that the management system should take in response to Sidecar lifecycle events.
Cannot be updated.

Type

object

Property	Type	Description
<code>postStart</code>	<code>object</code>	PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the

Property	Type	Description
		container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗
preStop	object	PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

.spec.sidecars[].lifecycle.postStart

Description

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
exec	object	Exec specifies a command to execute in the container.

Property	Type	Description
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to perform.
<code>sleep</code>	<code>object</code>	Sleep represents a duration that the container should sleep.
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

`.spec.sidecars[].lifecycle.postStart.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].lifecycle.postStart.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

`.spec.sidecars[].lifecycle.postStart.exec.command[]`

Type

string

`.spec.sidecars[].lifecycle.postStart.httpGet`

Description

HTTPGet specifies an HTTP GET request to perform.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.

Property	Type	Description
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>	<code>int</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].lifecycle.postStart.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.sidecars[].lifecycle.postStart.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.sidecars[].lifecycle.postStart.sleep`

Description

Sleep represents a duration that the container should sleep.

Type

`object`

Required

`seconds`

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.spec.sidecars[].lifecycle.postStart.tcpSocket`

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

Type

`object`

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.sidecars[].lifecycle.preStop

Description

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
exec	object	Exec specifies a command to execute in the container.

Property	Type	Description
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to perform.
<code>sleep</code>	<code>object</code>	Sleep represents a duration that the container should sleep.
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

`.spec.sidecars[].lifecycle.preStop.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].lifecycle.preStop.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.sidecars[].lifecycle.preStop.exec.command[]

Type

string

.spec.sidecars[].lifecycle.preStop.httpGet

Description

HTTPGet specifies an HTTP GET request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.

Property	Type	Description
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>	<code>int</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].lifecycle.preStop.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.sidecars[].lifecycle.preStop.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.sidecars[].lifecycle.preStop.sleep`

Description

Sleep represents a duration that the container should sleep.

Type

`object`

Required

`seconds`

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.spec.sidecars[].lifecycle.preStop.tcpSocket`

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

Type

`object`

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.sidecars[].livenessProbe

Description

Periodic probe of Sidecar liveness. Container will be restarted if the probe fails. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

object

Property	Type	Description
exec	object	Exec specifies a command to execute in the
failureThreshold	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.

Property	Type	Description
<code>grpc</code>	<code>object</code>	GRPC specifies a GRPC HealthCheckRequest
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies a connection to a TCP
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown on probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It must be longer than the expected cleanup time for the process. If this value is nil, the pod's terminationGracePeriodSeconds will be used

Property	Type	Description
		Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used when unset.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More information: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.sidecars[].livenessProbe.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].livenessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.sidecars[].livenessProbe.exec.command[]`

Type

`string`

`.spec.sidecars[].livenessProbe.grpc`

Description

GRPC specifies a GRPC HealthCheckRequest.

Type

`object`

Required

`port`

Property	Type	Description
<code>port</code>	<code>integer</code>	Port number of the gRPC service. Number must be in the range 1 to 65535.

Property	Type	Description
<code>service</code>	<code>string</code>	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

`.spec.sidecars[].livenessProbe.httpGet`

Description

HTTPGet specifies an HTTP GET request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.

Property	Type	Description
<code>port</code>	<code>integer</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].livenessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.sidecars[].livenessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name` `value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

Property	Type	Description
<code>value</code>	<code>string</code>	The header field value

`.spec.sidecars[].livenessProbe.tcpSocket`

Description

TCPSocket specifies a connection to a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.sidecars[].ports`

Description

List of ports to expose from the Sidecar. Exposing a port here gives the system additional information about the network connections a container uses, but is primarily informational. Not specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default "0.0.0.0" address inside a container will be accessible from the network. Cannot be updated.

Type

array

.spec.sidecars[].ports[]

Description

ContainerPort represents a network port in a single container.

Type

object

Required

containerPort

Property	Type	Description
containerPort	integer	Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.
hostIP	string	What host IP to bind the external port to.
hostPort	integer	Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.
name	string	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.

Property	Type	Description
<code>protocol</code>	<code>string</code>	Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

`.spec.sidecars[].readinessProbe`

Description

Periodic probe of Sidecar service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

`object`

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies a command to execute in the
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies a GRPC HealthCheckRequest to the service.
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to the service.
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Property	Type	Description
		https://kubernetes.io/docs/concepts/workload-lifecycle#container-probes ↗
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies a connection to a TCP endpoint.
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown on probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It specifies that the pod needs to terminate gracefully. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used if specified.

Property	Type	Description
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.sidecars[].readinessProbe.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].readinessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.sidecars[].readinessProbe.exec.command[]**Type**

string

.spec.sidecars[].readinessProbe.grpc**Description**

GRPC specifies a GRPC HealthCheckRequest.

Type

object

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.sidecars[].readinessProbe.httpGet

Description

HTTPGet specifies an HTTP GET request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
scheme	string	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.sidecars[].readinessProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

array

.spec.sidecars[].readinessProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

object

Required

name

value

Property	Type	Description
name	string	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
value	string	The header field value

.spec.sidecars[].readinessProbe.tcpSocket

Description

TCPSocket specifies a connection to a TCP port.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.sidecars[].securityContext`

Description

SecurityContext defines the security options the Sidecar should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

`object`

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: - run as Privileged - has <code>CAP_SYS_ADMIN</code> Note that this field cannot be set when <code>spec.os.name</code> is windows.

Property	Type	Description
<code>appArmorProfile</code>	<code>object</code>	<code>appArmorProfile</code> is the AppArmor options to use by this container. If set, this profile overrides the pod's <code>appArmorProfile</code> . Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>procMount</code>	<code>string</code>	<code>procMount</code> denotes the type of proc mount to use for the containers. The default value is <code>Default</code> which uses the container runtime defaults for readonly paths and masked paths. This requires the <code>ProcMountType</code> feature flag to be enabled. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this

Property	Type	Description
		field cannot be set when spec.os.name is windows.
runAsGroup	integer	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
runAsNonRoot	boolean	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

Property	Type	Description
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and

Property	Type	Description
		PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.sidecars[].securityContext.appArmorProfile`

Description

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

Type

object

Required

type

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.spec.sidecars[].securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

object

Property	Type	Description
<code>add</code>	array	Added capabilities
<code>drop</code>	array	Removed capabilities

`.spec.sidecars[].securityContext.capabilities.add`

Description

Added capabilities

Type

array

`.spec.sidecars[].securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.sidecars[].securityContext.capabilities.drop`

Description

Removed capabilities

Type

array

.spec.sidecars[].securityContext.capabilities.drop[]

Description

Capability represent POSIX capabilities type

Type

string

.spec.sidecars[].securityContext.seLinuxOptions

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
level	string	Level is SELinux level label that applies to the container.
role	string	Role is a SELinux role label that applies to the container.

Property	Type	Description
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.spec.sidecars[].securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when `spec.os.name` is `windows`.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	<code>localhostProfile</code> indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
<code>type</code>	<code>string</code>	<code>type</code> indicates which kind of seccomp profile will be applied. Valid options are:

Property	Type	Description
		Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.spec.sidecars[].securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
gmsaCredentialSpec	string	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
gmsaCredentialSpecName	string	GMSACredentialSpecName is the name of the GMSA credential spec to use.
hostProcess	boolean	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective

Property	Type	Description
		HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
runAsUserName	string	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

.spec.sidecars[].startupProbe

Description

StartupProbe indicates that the Pod the Sidecar is running in has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

object

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies a command to execute in the
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies a GRPC HealthCheckRequest to the service.
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to the container's port.
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies a connection to a TCP port.

Property	Type	Description
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs gracefully upon probe failure. The grace period duration in seconds after the processes running in the pod are sent a termination signal and the timer begins. Once the grace period has elapsed, the processes are forcibly halted with a kill signal. A value greater than zero indicates a grace period; a value of zero indicates no grace period. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.sidecars[].startupProbe.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].startupProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.sidecars[].startupProbe.exec.command[]`

Type

`string`

`.spec.sidecars[].startupProbe.grpc`

Description

GRPC specifies a GRPC HealthCheckRequest.

Type

`object`

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.sidecars[].startupProbe.httpGet

Description

HTTPGet specifies an HTTP GET request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

Property	Type	Description
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].startupProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.sidecars[].startupProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.sidecars[].startupProbe.tcpSocket`

Description

TCPSocket specifies a connection to a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.sidecars[].volumeDevices`

Description

volumeDevices is the list of block devices to be used by the Sidecar.

Type

array

.spec.sidecars[].volumeDevices[]

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

object

Required

devicePath

name

Property	Type	Description
devicePath	string	devicePath is the path inside of the container that the device will be mapped to.
name	string	name must match the name of a persistentVolumeClaim in the pod

.spec.sidecars[].volumeMounts

Description

Volumes to mount into the Sidecar's filesystem. Cannot be updated.

Type

array

.spec.sidecars[].volumeMounts[]

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
mountPath	string	Path within the container at which the volume should be mounted. Must not contain ':'. mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10. When RecursiveReadOnly is set to IfPossible or to Enabled, MountPropagation must be None or unspecified (which defaults to None).
name	string	This must match the Name of a Volume.
readOnly	boolean	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
recursiveReadOnly	string	RecursiveReadOnly specifies whether read-only mounts should be handled recursively. If ReadOnly is false, this field has no meaning and must be unspecified.

Property	Type	Description
		If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

`.spec.sidecars[].workspaces`

Description

This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Sidecar wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

Type

array

`.spec.sidecars[].workspaces[]`

Description

WorkspaceUsage is used by a Step or Sidecar to declare that it wants isolated access to a Workspace defined in a Task.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath is the path that the workspace should be mounted to inside the Step or Sidecar, overriding any MountPath specified in the Task's WorkspaceDeclaration.
<code>name</code>	<code>string</code>	Name is the name of the workspace this Step or Sidecar wants access to.

`.spec.stepTemplate`

Description

StepTemplate can be used as the basis for all step containers within the Task, so that the steps inherit settings on the base container.

Type

object

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell

Property	Type	Description
<code>computeResources</code>	<code>object</code>	ComputeResources required by this Step. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>env</code>	<code>array</code>	List of environment variables to set in the Step. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.
<code>image</code>	<code>string</code>	Image reference name. More info: https://kubernetes.io/docs/concepts/containers/images ↗
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Default to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#update-images ↗
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Step should run with. If set, the fields of SecurityContext override the

Property	Type	Description
		equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>volumeDevices</code>	<code>array</code>	<code>volumeDevices</code> is the list of block devices to be used by the Step.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Step's filesystem. Cannot be upda
<code>workingDir</code>	<code>string</code>	Step's working directory. If not specified, the container runtime default will be used, which might be configured in the contain image. Cannot be updated.

`.spec.stepTemplate.args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

`array`

`.spec.stepTemplate.args[]`

Type

string

.spec.stepTemplate.command

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.spec.stepTemplate.command[]

Type

string

.spec.stepTemplate.computeResources

Description

ComputeResources required by this Step. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
claims	array	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.stepTemplate.computeResources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

.spec.stepTemplate.computeResources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
name	string	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.
request	string	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

.spec.stepTemplate.computeResources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.stepTemplate.computeResources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.stepTemplate.env

Description

List of environment variables to set in the Step. Cannot be updated.

Type

array

.spec.stepTemplate.env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.
value	string	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged.

Property	Type	Description
		Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

.spec.stepTemplate.env[].valueFrom

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory,

Property	Type	Description
		limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.stepTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.stepTemplate.env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.stepTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (`limits.cpu`, `limits.memory`, `limits.ephemeral-storage`, `requests.cpu`, `requests.memory` and `requests.ephemeral-storage`) are currently supported.

Type

`object`**Required**`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.stepTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type`object`**Required**`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.stepTemplate.envFrom`

Description

List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

`.spec.stepTemplate.envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.spec.stepTemplate.envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.spec.stepTemplate.envFrom[].secretRef`

Description

The Secret to select from

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

`.spec.stepTemplate.securityContext`

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info:
<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

object

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its

Property	Type	Description
		parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. <code>AllowPrivilegeEscalation</code> is true always when the container is: - run as Privileged - has <code>CAP_SYS_ADMIN</code> Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>appArmorProfile</code>	<code>object</code>	<code>appArmorProfile</code> is the AppArmor options to use by this container. If set, this profile overrides the pod's <code>appArmorProfile</code>. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>procMount</code>	<code>string</code>	<code>procMount</code> denotes the type of proc mount to use for the containers. The default value is

Property	Type	Description
		Default which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.stepTemplate.securityContext.appArmorProfile`

Description

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.spec.stepTemplate.securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

`.spec.stepTemplate.securityContext.capabilities.add`

Description

Added capabilities

Type

array

`.spec.stepTemplate.securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.stepTemplate.securityContext.capabilities.drop`

Description

Removed capabilities

Type

array

`.spec.stepTemplate.securityContext.capabilities.drop[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.stepTemplate.securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value

specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
level	string	Level is SELinux level label that applies to the container.
role	string	Role is a SELinux role label that applies to the container.
type	string	Type is a SELinux type label that applies to the container.
user	string	User is a SELinux user label that applies to the container.

.spec.stepTemplate.securityContext.seccompProfile

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

Type

object

Required

type

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
<code>type</code>	<code>string</code>	type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

`.spec.stepTemplate.securityContext.windowsOptions`

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

`object`

Property	Type	Description
<code>gmsaCredentialSpec</code>	<code>string</code>	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-

Property	Type	Description
		gmsa ) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
<code>gmsaCredentialSpecName</code>	<code>string</code>	<code>GMSACredentialSpecName</code> is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	<code>HostProcess</code> determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective <code>HostProcess</code> value (it is not allowed to have a mix of <code>HostProcess</code> containers and non- <code>HostProcess</code> containers). In addition, if <code>HostProcess</code> is true then <code>HostNetwork</code> must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The <code>UserName</code> in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in <code>PodSecurityContext</code> . If set in both <code>SecurityContext</code> and <code>PodSecurityContext</code> , the value specified in <code>SecurityContext</code> takes precedence.

`.spec.stepTemplate.volumeDevices`

Description

`volumeDevices` is the list of block devices to be used by the Step.

Type

array

.spec.stepTemplate.volumeDevices[]

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

object

Required

devicePath

name

Property	Type	Description
devicePath	string	devicePath is the path inside of the container that the device will be mapped to.
name	string	name must match the name of a persistentVolumeClaim in the pod

.spec.stepTemplate.volumeMounts

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type

array

.spec.stepTemplate.volumeMounts[]

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
mountPath	string	Path within the container at which the volume should be mounted. Must not contain ':'. mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10. When RecursiveReadOnly is set to IfPossible or to Enabled, MountPropagation must be None or unspecified (which defaults to None).
name	string	This must match the Name of a Volume.
readOnly	boolean	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
recursiveReadOnly	string	RecursiveReadOnly specifies whether read-only mounts should be handled recursively. If ReadOnly is false, this field has no meaning and must be unspecified.

Property	Type	Description
		If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

.spec.steps

Description

Steps are the steps of the build; each step is run sequentially with the source mounted into /workspace.

Type

array

.spec.steps[]

Description

Step runs a subcomponent of a Task

Type

object

Required

name

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input str

Property	Type	Description
		will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
<code>computeResources</code>	<code>object</code>	ComputeResources required by this Step. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/
<code>env</code>	<code>array</code>	List of environment variables to set in the Step. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.
<code>image</code>	<code>string</code>	OCI image name. More info: https://kubernetes.io/docs/concepts/containers/images

Property	Type	Description
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Default to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#updating-images ↗
<code>name</code>	<code>string</code>	Name of the Step specified as a DNS_LABEL. Each Step in a Task must have a unique name.
<code>onError</code>	<code>string</code>	OnError defines the exiting behavior of a container on error can be set to [continue stopAndFail]
<code>params</code>	<code>array</code>	Params declares parameters passed to this step action.
<code>ref</code>	<code>object</code>	Contains the reference to an existing StepAction.
<code>results</code>	<code>array</code>	Results declares StepResults produced by the Step. It can be used in an inlined Step when used to store Results with <code>\$(step.results.resultName.path)</code> . It cannot be used when referencing StepActions using [v1.Step.Ref]. The Results declared by the StepActions will be stored here instead.
<code>script</code>	<code>string</code>	Script is the contents of an executable file to execute.

Property	Type	Description
		If Script is not empty, the Step cannot have an Command and the Args will be passed to the Script.
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Step should run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>stderrConfig</code>	<code>object</code>	Stores configuration for the stderr stream of the step.
<code>stdoutConfig</code>	<code>object</code>	Stores configuration for the stdout stream of the step.
<code>timeout</code>	<code>string</code>	Timeout is the time after which the step times out. Defaults to never. Refer to Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration ↗
<code>volumeDevices</code>	<code>array</code>	volumeDevices is the list of block devices to be used by the Step.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Step's filesystem. Cannot be updated.
<code>when</code>	<code>array</code>	When is a list of when expressions that need to be true for the task to run

Property	Type	Description
<code>workingDir</code>	<code>string</code>	Step's working directory. If not specified, the container runtime default will be used, which might be configured in the container image. Cannot be updated.
<code>workspaces</code>	<code>array</code>	This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Step wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

`.spec.steps[].args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e.

`$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated.

More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

`array`

`.spec.steps[].args[]`

Type

`string`

`.spec.steps[].command`

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.spec.steps[].command[]`

Type

string

`.spec.steps[].computeResources`

Description

ComputeResources required by this Step. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
<code>claims</code>	array	Claims lists the names of resources, defined in <code>spec.resourceClaims</code>, that are used by this container. This is an alpha field and requires enabling the <code>DynamicResourceAllocation</code> feature gate.

Property	Type	Description
		This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.steps[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.spec.steps[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
name	string	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.
request	string	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

.spec.steps[].computeResources.limits**Description**

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.steps[].computeResources.requests**Description**

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.steps[].env`

Description

List of environment variables to set in the Step. Cannot be updated.

Type

array

`.spec.steps[].env[]`

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the environment variable. Must be a C_IDENTIFIER.
<code>value</code>	<code>string</code>	Variable references <code>\$(VAR_NAME)</code> are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to <code>""</code>.

Property	Type	Description
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.steps[].env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code> , <code>metadata.namespace</code> , <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , <code>spec.nodeName</code> , <code>spec.serviceAccountName</code> , <code>status.hostIP</code> , <code>status.podIP</code> , <code>status.podIPs</code> .
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (<code>limits.cpu</code> , <code>limits.memory</code> , <code>limits.ephemeral-storage</code> , <code>requests.cpu</code> , <code>requests.memory</code> and <code>requests.ephemeral-storage</code>) are currently supported.

Property	Type	Description
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.steps[].env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.steps[].env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

.spec.steps[].env[].valueFrom.resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.steps[].env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.steps[].envFrom`

Description

List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

`.spec.steps[].envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from

Property	Type	Description
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.spec.steps[].envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.spec.steps[].envFrom[].secretRef`

Description

The Secret to select from

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	Specify whether the Secret must be defined

.spec.steps[].params

Description

Params declares parameters passed to this step action.

Type

array

.spec.steps[].params[]

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

`.spec.steps[].ref`

Description

Contains the reference to an existing StepAction.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referenced step
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.steps[].ref.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

array

.spec.steps[].ref.params[]

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

.spec.steps[].results

Description

Results declares StepResults produced by the Step. It can be used in an inlined Step when used to store Results to \$(step.results.resultName.path). It cannot be used when referencing StepActions using [v1.Step.Ref]. The Results declared by the StepActions will be stored here instead.

Type

array

.spec.steps[].results[]

Description

StepResult used to describe the Results of a Step.

Type

`object`**Required**`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a human-readable description of the result
<code>name</code>	<code>string</code>	Name the given name
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs results.
<code>type</code>	<code>string</code>	The possible types are 'string', 'array', and 'object', with 'string' as the default.

`.spec.steps[].results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type`object`

`.spec.steps[].securityContext`

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

object

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: 1. run as Privileged 2. has CAP_SYS_ADMIN Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>appArmorProfile</code>	<code>object</code>	appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to

Property	Type	Description
		false. Note that this field cannot be set when spec.os.name is windows.
procMount	string	procMount denotes the type of proc mount to use for the containers. The default value is Default which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
readOnlyRootFilesystem	boolean	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.
runAsGroup	integer	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
runAsNonRoot	boolean	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to

Property	Type	Description
		start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
runAsUser	integer	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
seLinuxOptions	object	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
seccompProfile	object	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that

Property	Type	Description
		this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.steps[].securityContext.appArmorProfile`

Description

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".

Property	Type	Description
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.spec.steps[].securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

`.spec.steps[].securityContext.capabilities.add`

Description

Added capabilities

Type

`array`

`.spec.steps[].securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.steps[].securityContext.capabilities.drop`

Description

Removed capabilities

Type

array

`.spec.steps[].securityContext.capabilities.drop[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.steps[].securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.spec.steps[].securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when `spec.os.name` is `windows`.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	<code>localhostProfile</code> indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured

Property	Type	Description
		seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
		type indicates which kind of seccomp profile will be applied. Valid options are:
type	string	Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.spec.steps[].securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
gmsaCredentialSpec	string	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the GMSACredentialSpecName field.

Property	Type	Description
<code>gmsaCredentialSpecName</code>	<code>string</code>	GMSACredentialSpecName is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

`.spec.steps[].stderrConfig`

Description

Stores configuration for the stderr stream of the step.

Type

`object`

Property	Type	Description
<code>path</code>	<code>string</code>	Path to duplicate stdout stream to on container's local filesystem.

`.spec.steps[].stdoutConfig`

Description

Stores configuration for the stdout stream of the step.

Type

`object`

Property	Type	Description
<code>path</code>	<code>string</code>	Path to duplicate stdout stream to on container's local filesystem.

`.spec.steps[].volumeDevices`

Description

volumeDevices is the list of block devices to be used by the Step.

Type

`array`

`.spec.steps[].volumeDevices[]`

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

`object`

Required

devicePath

name

Property	Type	Description
devicePath	string	devicePath is the path inside of the container that the device will be mapped to.
name	string	name must match the name of a persistentVolumeClaim in the pod

.spec.steps[].volumeMounts

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type

array

.spec.steps[].volumeMounts[]

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'.
<code>mountPropagation</code>	<code>string</code>	<code>mountPropagation</code> determines how mounts are propagated from the host to container and the other way around. When not set, <code>MountPropagationNone</code> is used. This field is beta in 1.10. When <code>RecursiveReadOnly</code> is set to <code>IfPossible</code> or to <code>Enabled</code>, <code>MountPropagation</code> must be <code>None</code> or unspecified (which defaults to <code>None</code>).
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>recursiveReadOnly</code>	<code>string</code>	<code>RecursiveReadOnly</code> specifies whether read-only mounts should be handled recursively. If <code>ReadOnly</code> is false, this field has no meaning and must be unspecified. If <code>ReadOnly</code> is true, and this field is set to <code>Disabled</code>, the mount is not made recursively read-only. If this field is set to <code>IfPossible</code>, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to <code>Enabled</code>, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not

Property	Type	Description
		be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

`.spec.steps[].when`

Description

When is a list of when expressions that need to be true for the task to run

Type

`array`

`.spec.steps[].when[]`

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

object

Property	Type	Description
<code>cel</code>	<code>string</code>	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md ↗
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.spec.steps[].when[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

array

`.spec.steps[].when[].values[]`

Type

`string`

`.spec.steps[].workspaces`

Description

This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Step wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

Type

`array`

`.spec.steps[].workspaces[]`

Description

WorkspaceUsage is used by a Step or Sidecar to declare that it wants isolated access to a Workspace defined in a Task.

Type

`object`

Required

`mountPath``name`

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath is the path that the workspace should be mounted to inside the Step or Sidecar, overriding any MountPath specified in the Task's WorkspaceDeclaration.

Property	Type	Description
<code>name</code>	<code>string</code>	Name is the name of the workspace this Step or Sidecar wants access to.

`.spec.workspaces`

Description

Workspaces are the volumes that this Task requires.

Type

`array`

`.spec.workspaces[]`

Description

WorkspaceDeclaration is a declaration of a volume that a Task requires.

Type

`object`

Required

`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is an optional human readable description of this volume.
<code>mountPath</code>	<code>string</code>	MountPath overrides the directory that the volume will be made available at.

Property	Type	Description
<code>name</code>	<code>string</code>	Name is the name by which you can bind the volume at runtime.
<code>optional</code>	<code>boolean</code>	Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.
<code>readOnly</code>	<code>boolean</code>	ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1/namespaces/{namespace}/tasks`
 - `DELETE` : delete collection of Task
 - `GET` : list objects of kind Task
 - `POST` : create a new Task
- `/apis/tekton.dev/v1/namespaces/{namespace}/tasks/{name}`
 - `DELETE` : delete the specified Task
 - `GET` : read the specified Task
 - `PATCH` : partially update the specified Task
 - `PUT` : replace the specified Task

/apis/tekton.dev/v1/namespaces/{namespace}/tasks

HTTP method

[DELETE](#)

Description

delete collection of Task

HTTP responses

HTTP code	Response body
200 - OK	Status ✓ schema
401 - Unauthorized	Empty

HTTP method

[GET](#)

Description

list objects of kind Task

HTTP responses

HTTP code	Response body
200 - OK	TaskList schema
401 - Unauthorized	Empty

HTTP method

[POST](#)

Description

create a new Task

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Task</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>Task</code> schema
201 - Created	<code>Task</code> schema
202 - Accepted	<code>Task</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

/apis/tekton.dev/v1/namespaces/{namespace}/tasks/{name}

HTTP method

[DELETE](#)

Description

delete the specified Task

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
202 - Accepted	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

[GET](#)

Description

read the specified Task

HTTP responses

HTTP code	Response body
200 - OK	<code>Task</code> schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Task

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>Task</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified Task

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Task</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>Task</code> schema
201 - Created	<code>Task</code> schema
401 - Unauthorized	Empty

PipelineRun [tekton.dev/v1]

Description

PipelineRun represents a single execution of a Pipeline. PipelineRuns are how the graph of Tasks declared in a Pipeline are executed; they specify inputs to Pipelines such as parameter values and capture operational aspects of the Tasks execution such as service account and tolerations. Creating a PipelineRun creates TaskRuns for Tasks in the referenced Pipeline.

Type

`object`

Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
<code>metadata</code>	<code>ObjectMeta</code> ↗	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	PipelineRunSpec defines the desired state of PipelineRun
<code>status</code>	<code>object</code>	PipelineRunStatus defines the observed state of PipelineRun

.spec

Description

PipelineRunSpec defines the desired state of PipelineRun

Type

`object`

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of parameter names and values.
<code>pipelineRef</code>	<code>object</code>	PipelineRef can be used to refer to a specific instance of a Pipeline.

Property	Type	Description
<code>pipelineSpec</code>		Specifying PipelineSpec can be disabled by setting <code>disable-inline-spec</code> feature flag. See Pipeline.spec (API version: tekton.dev/v1)
<code>status</code>	<code>string</code>	Used for cancelling a pipelinerun (and maybe more later on)
<code>taskRunSpecs</code>	<code>array</code>	TaskRunSpecs holds a set of runtime specs
<code>taskRunTemplate</code>	<code>object</code>	TaskRunTemplate represent template of taskrun
<code>timeouts</code>	<code>object</code>	Time after which the Pipeline times out. Currently three keys are accepted in the map pipeline, tasks and finally with <code>Timeouts.pipeline >= Timeouts.tasks + Timeouts.finally</code>
<code>workspaces</code>	<code>array</code>	Workspaces holds a set of workspace bindings that must match names with those declared in the pipeline.

.spec.params

Description

Params is a list of parameter names and values.

Type

`array`

.spec.params[]

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

.spec.pipelineRef

Description

PipelineRef can be used to refer to a specific instance of a Pipeline.

Type

object

Property	Type	Description
apiVersion	string	API version of the referent
name	string	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names
params	array	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include

Property	Type	Description
		"repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.pipelineRef.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.pipelineRef.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

.spec.taskRunSpecs

Description

TaskRunSpecs holds a set of runtime specs

Type

array

.spec.taskRunSpecs[]

Description

PipelineTaskRunSpec can be used to configure specific specs for a concrete Task

Type

object

Property	Type	Description
<code>computeResources</code>	<code>object</code>	Compute resources to use for this TaskRun
<code>metadata</code>	<code>ObjectMeta</code> ↗	PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask
<code>pipelineTaskName</code>	<code>string</code>	
<code>podTemplate</code>	<code>object</code>	PodTemplate holds pod specific configuration
<code>serviceAccountName</code>	<code>string</code>	
<code>sidecarSpecs</code>	<code>array</code>	
<code>stepSpecs</code>	<code>array</code>	

`.spec.taskRunSpecs[].computeResources`

Description

Compute resources to use for this TaskRun

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in <code>spec.resourceClaims</code> , that are used by this container.
<code>claims</code>	<code>array</code>	This is an alpha field and requires enabling the <code>DynamicResourceAllocation</code> feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.taskRunSpecs[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.spec.taskRunSpecs[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.spec.taskRunSpecs[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.taskRunSpecs[].computeResources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.taskRunSpecs[].podTemplate`

Description

PodTemplate holds pod specific configuration

Type

object

Property	Type	Description
<code>affinity</code>	<code>object</code>	If specified, the pod's scheduling constraints. Pod.spec.affinity (API version: v1)
<code>automountServiceAccountToken</code>	<code>boolean</code>	AutomountServiceAccountToken indicates whether the pod is running as this service account should have a token automatically mounted.
<code>dnsConfig</code>	<code>object</code>	Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated Pod.spec.dnsConfig.

Property	Type	Description
		configuration based on DNSPolicy.
<code>dnsPolicy</code>	<code>string</code>	Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. Parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.
<code>enableServiceLinks</code>	<code>boolean</code>	EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Defaults to true.
<code>env</code>	<code>array</code>	List of environment variables that can be provided to containers belonging to the pod.
<code>hostAliases</code>	<code>array</code>	HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.
<code>hostNetwork</code>	<code>boolean</code>	HostNetwork specifies whether the pod may use the host's network namespace.
<code>imagePullSecrets</code>	<code>array</code>	ImagePullSecrets gives the name of the secret the pod to pull the image if specified.

Property	Type	Description
<code>nodeSelector</code>	<code>object</code>	NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match labels for the pod to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/node-selector/
<code>priorityClassName</code>	<code>string</code>	If specified, indicates the pod's priority. "system-cluster-critical" and "system-node-critical" are two special keywords which indicate the highest priorities within the namespace; the former being the highest priority. Any other name can be used to indicate a lower priority. Priority classes can be defined by creating a PriorityClass object with the appropriate name. If not specified, the pod priority will be default, which is zero if there is no default.
<code>runtimeClassName</code>	<code>string</code>	RuntimeClassName refers to a RuntimeClass object in the node.k8s.io group, which should be used to run this pod. If no RuntimeClass resource matches the pod specification, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is a class with an empty definition that uses the default runtime handler. More info: https://git.k8s.io/enhancements/keps/sig-node/runtime-class.md This is a beta feature as of Kubernetes v1.14.
<code>schedulerName</code>	<code>string</code>	SchedulerName specifies the scheduler to be used to dispatch the Pod

Property	Type	Description
<code>securityContext</code>		SecurityContext holds pod-level security attributes and common container settings. Optional: Default: nil. See type description for default values of each field. Pod.spec.securityContext (API version: v1)
<code>tolerations</code>	<code>array</code>	If specified, the pod's tolerations.
<code>topologySpreadConstraints</code>	<code>array</code>	TopologySpreadConstraints controls how Pods spread across your cluster among failure domains such as regions, zones, nodes, and other user-defined topology domains.
<code>volumes</code>		List of volumes that can be mounted by containers belonging to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes See Pod.spec.volumes (API version: v1)

`.spec.taskRunSpecs[].podTemplate.dnsConfig`

Description

Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

Type

`object`

Property	Type	Description
<code>nameservers</code>	<code>array</code>	A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.
<code>options</code>	<code>array</code>	A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.
<code>searches</code>	<code>array</code>	A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

`.spec.taskRunSpecs[].podTemplate.dnsConfig.nameservers`

Description

A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Type

`array`

`.spec.taskRunSpecs[].podTemplate.dnsConfig.nameservers[]`

Type

`string`

`.spec.taskRunSpecs[].podTemplate.dnsConfig.options`

Description

A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.

Type

array

`.spec.taskRunSpecs[].podTemplate.dnsConfig.options[]`

Description

PodDNSConfigOption defines DNS resolver options of a pod.

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name is this DNS resolver option's name. Required.
<code>value</code>	<code>string</code>	Value is this DNS resolver option's value.

`.spec.taskRunSpecs[].podTemplate.dnsConfig.searches`

Description

A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

Type

array

`.spec.taskRunSpecs[].podTemplate.dnsConfig.searches[]`

Type

`string`

`.spec.taskRunSpecs[].podTemplate.env`

Description

List of environment variables that can be provided to the containers belonging to the pod.

Type

`array`

`.spec.taskRunSpecs[].podTemplate.env[]`

Description

EnvVar represents an environment variable present in a Container.

Type

`object`

Required

`name`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the environment variable. Must be a C_IDENTIFIER.
<code>value</code>	<code>string</code>	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce

Property	Type	Description
		the string literal "\${VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.taskRunSpecs[].podTemplate.env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu,

Property	Type	Description
		requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.taskRunSpecs[].podTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.taskRunSpecs[].podTemplate.env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels[<KEY>]``, ``metadata.annotations[<KEY>]``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.taskRunSpecs[].podTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.taskRunSpecs[].podTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.taskRunSpecs[].podTemplate.hostAliases`

Description

HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.

Type

`array`

`.spec.taskRunSpecs[].podTemplate.hostAliases[]`

Description

HostAlias holds the mapping between IP and hostnames that will be injected as an entry in the pod's hosts file.

Type

`object`

Required

ip

Property	Type	Description
hostnames	array	Hostnames for the above IP address.
ip	string	IP address of the host file entry.

`.spec.taskRunSpecs[].podTemplate.hostAliases[].hostnames`

Description

Hostnames for the above IP address.

Type

array

`.spec.taskRunSpecs[].podTemplate.hostAliases[].hostnames[]`

Type

string

`.spec.taskRunSpecs[].podTemplate.imagePullSecrets`

Description

ImagePullSecrets gives the name of the secret used by the pod to pull the image if specified

Type

array

`.spec.taskRunSpecs[].podTemplate.imagePullSecrets[]`

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names

`.spec.taskRunSpecs[].podTemplate.nodeSelector`

Description

NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info: <https://kubernetes.io/docs/concepts/configuration/assign-pod-node/>

Type

object

`.spec.taskRunSpecs[].podTemplate.tolerations`

Description

If specified, the pod's tolerations.

Type

array

.spec.taskRunSpecs[].podTemplate.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	string	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	string	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	string	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
tolerationSeconds	integer	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint

Property	Type	Description
		forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
<code>value</code>	<code>string</code>	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints`

Description

TopologySpreadConstraints controls how Pods are spread across your cluster among failure-domains such as regions, zones, nodes, and other user-defined topology domains.

Type

`array`

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[]`

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

`object`

Required

`maxSkew`

`topologyKey`

`whenUnsatisfiable`

Property	Type	Description
<code>labelSelector</code>	<code>object</code>	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.
<code>matchLabelKeys</code>	<code>array</code>	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).
<code>maxSkew</code>	<code>integer</code>	MaxSkew describes the degree to which pods may be unevenly distributed. When <code>whenUnsatisfiable=DoNotSchedule</code>, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone

Property	Type	Description
		cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. zone1 zone2 zone3 P P P P P - if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). - if MaxSkew is 2, incoming pod can be scheduled onto any zone. When <code>whenUnsatisfiable=ScheduleAnyway</code>, it is used to give higher precedence to topologies that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.
<code>minDomains</code>	<code>integer</code>	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule.

Property	Type	Description
		For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: zone1 zone2 zone3 P P P P P P The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.
nodeAffinityPolicy	string	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are: - Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. - Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations. If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
nodeTaintsPolicy	string	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are: Honor: nodes without taints, along with tainted nodes for which the incoming pod has a

Property	Type	Description
		toleration, are included. Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
<code>topologyKey</code>	<code>string</code>	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It's a required field.
<code>whenUnsatisfiable</code>	<code>string</code>	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. - DoNotSchedule (default) tells the scheduler not to schedule it. - ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher

Property	Type	Description
		precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: zone1 zone2 zone3 P P P P P If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it <i>more</i> imbalanced. It's a required field.

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].labelSelector`

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
<code>matchLabels</code>	<code>object</code>	<code>matchLabels</code> is a map of {key,value} pairs. A single {key,value} in the <code>matchLabels</code> map is equivalent to an element of <code>matchExpressions</code> , whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions`

Description

`matchExpressions` is a list of label selector requirements. The requirements are ANDed.

Type

`array`

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key``operator`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the label key that the selector applies to.
<code>operator</code>	<code>string</code>	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
<code>values</code>	<code>array</code>	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values[]`

Type

`string`

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].labelSelector.matchLabels`

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].matchLabelKeys`

Description

MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

Type

array

`.spec.taskRunSpecs[].podTemplate.topologySpreadConstraints[].matchLabelKeys[]`

Type

string

`.spec.taskRunSpecs[].sidecarSpecs`

Type

array

.spec.taskRunSpecs[].sidecarSpecs[]

Description

TaskRunSidecarSpec is used to override the values of a Sidecar in the corresponding Task.

Type

object

Required

computeResources

name

Property	Type	Description
computeResources	object	The resource requirements to apply to the Sidecar.
name	string	The name of the Sidecar to override.

.spec.taskRunSpecs[].sidecarSpecs[].computeResources

Description

The resource requirements to apply to the Sidecar.

Type

object

Property	Type	Description
claims	array	Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.

Property	Type	Description
		This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.taskRunSpecs[].sidecarSpecs[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

`array`

`.spec.taskRunSpecs[].sidecarSpecs[].computeResources.claims[]`

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.spec.taskRunSpecs[].sidecarSpecs[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.taskRunSpecs[].sidecarSpecs[].computeResources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.taskRunSpecs[].stepSpecs`

Type

array

`.spec.taskRunSpecs[].stepSpecs[]`

Description

TaskRunStepSpec is used to override the values of a Step in the corresponding Task.

Type

object

Required

computeResources

name

Property	Type	Description
<code>computeResources</code>	<code>object</code>	The resource requirements to apply to the Step.
<code>name</code>	<code>string</code>	The name of the Step to override.

`.spec.taskRunSpecs[].stepSpecs[].computeResources`

Description

The resource requirements to apply to the Step.

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in <code>spec.resourceClaims</code> , that are used by this container.
<code>claims</code>	<code>array</code>	This is an alpha field and requires enabling the <code>DynamicResourceAllocation</code> feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.taskRunSpecs[].stepSpecs[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.spec.taskRunSpecs[].stepSpecs[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.spec.taskRunSpecs[].stepSpecs[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.taskRunSpecs[].stepSpecs[].computeResources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.taskRunTemplate`

Description

TaskRunTemplate represent template of taskrun

Type

object

Property	Type	Description
<code>podTemplate</code>	object	PodTemplate holds pod specific configuration

Property	Type	Description
<code>serviceAccountName</code>	<code>string</code>	

`.spec.taskRunTemplate.podTemplate`

Description

PodTemplate holds pod specific configuration

Type

`object`

Property	Type	Description
<code>affinity</code>		If specified, the pod's scheduling constraints. Pod.spec.affinity (API version: v1)
<code>automountServiceAccountToken</code>	<code>boolean</code>	AutomountServiceAccountToken indicates whether running as this service account should have a token automatically mounted.
<code>dnsConfig</code>	<code>object</code>	Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated configuration based on DNSPolicy.
<code>dnsPolicy</code>	<code>string</code>	Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. Parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

Property	Type	Description
<code>enableServiceLinks</code>	<code>boolean</code>	EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Defaults to true.
<code>env</code>	<code>array</code>	List of environment variables that can be provided to containers belonging to the pod.
<code>hostAliases</code>	<code>array</code>	HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.
<code>hostNetwork</code>	<code>boolean</code>	HostNetwork specifies whether the pod may use the host's network namespace
<code>imagePullSecrets</code>	<code>array</code>	ImagePullSecrets gives the name of the secret in the pod's namespace to use for pulling private images. This field is optional, and it is ignored if no secret is provided. It is only valid when pod security policies are enabled.
<code>nodeSelector</code>	<code>object</code>	NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/node-selector/

Property	Type	Description
<code>priorityClassName</code>	<code>string</code>	If specified, indicates the pod's priority. "system-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities in the namespace; they are not assigned by the system, but rather by the user. Any other name must be defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default, which is zero if there is no default.
<code>runtimeClassName</code>	<code>string</code>	RuntimeClassName refers to a RuntimeClass object in the node.k8s.io group, which should be used to run this pod. If no RuntimeClass resource matches the pod class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is a default class with an empty definition that uses the default runtime handler. More info: https://git.k8s.io/enhancements/keps/sig-node/runtime-class.md This is a beta feature as of Kubernetes v1.14.
<code>schedulerName</code>	<code>string</code>	SchedulerName specifies the scheduler to be used to dispatch the Pod
<code>securityContext</code>	<code>PodSecurityContext</code>	SecurityContext holds pod-level security attributes and common container settings. Optional: Default securityContext will be set if this field is not present. See type description for default values of each field. Pod.spec.securityContext (API version: v1)

Property	Type	Description
<code>tolerations</code>	<code>array</code>	If specified, the pod's tolerations.
<code>topologySpreadConstraints</code>	<code>array</code>	TopologySpreadConstraints controls how Pods spread across your cluster among failure-domains as regions, zones, nodes, and other user-defined topology domains.
<code>volumes</code>		List of volumes that can be mounted by containers belonging to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes See Pod.spec.volumes (API version: v1)

`.spec.taskRunTemplate.podTemplate.dnsConfig`

Description

Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

Type

`object`

Property	Type	Description
<code>nameservers</code>	<code>array</code>	A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Property	Type	Description
<code>options</code>	<code>array</code>	A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.
<code>searches</code>	<code>array</code>	A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

`.spec.taskRunTemplate.podTemplate.dnsConfig.nameservers`

Description

A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Type

`array`

`.spec.taskRunTemplate.podTemplate.dnsConfig.nameservers[]`

Type

`string`

`.spec.taskRunTemplate.podTemplate.dnsConfig.options`

Description

A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.

Type

array

`.spec.taskRunTemplate.podTemplate.dnsConfig.options[]`

Description

PodDNSConfigOption defines DNS resolver options of a pod.

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name is this DNS resolver option's name. Required.
<code>value</code>	<code>string</code>	Value is this DNS resolver option's value.

`.spec.taskRunTemplate.podTemplate.dnsConfig.searches`

Description

A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

Type

array

`.spec.taskRunTemplate.podTemplate.dnsConfig.searches[ ]`

Type

string

.spec.taskRunTemplate.podTemplate.env

Description

List of environment variables that can be provided to the containers belonging to the pod.

Type

array

.spec.taskRunTemplate.podTemplate.env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.
value	string	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never

Property	Type	Description
		be expanded, regardless of whether the variable exists or not. Defaults to "".
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.taskRunTemplate.podTemplate.env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Property	Type	Description
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.taskRunTemplate.podTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.taskRunTemplate.podTemplate.env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.taskRunTemplate.podTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.taskRunTemplate.podTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.taskRunTemplate.podTemplate.hostAliases`

Description

HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.

Type

`array`

`.spec.taskRunTemplate.podTemplate.hostAliases[]`

Description

HostAlias holds the mapping between IP and hostnames that will be injected as an entry in the pod's hosts file.

Type

`object`

Required

ip

Property	Type	Description
hostnames	array	Hostnames for the above IP address.
ip	string	IP address of the host file entry.

.spec.taskRunTemplate.podTemplate.hostAliases[].hostnames

Description

Hostnames for the above IP address.

Type

array

.spec.taskRunTemplate.podTemplate.hostAliases[].hostnames[]

Type

string

.spec.taskRunTemplate.podTemplate.imagePullSecrets

Description

ImagePullSecrets gives the name of the secret used by the pod to pull the image if specified

Type

array

.spec.taskRunTemplate.podTemplate.imagePullSecrets[]

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names

.spec.taskRunTemplate.podTemplate.nodeSelector

Description

NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info: <https://kubernetes.io/docs/concepts/configuration/assign-pod-node/>

Type

object

.spec.taskRunTemplate.podTemplate.tolerations

Description

If specified, the pod's tolerations.

Type

array

.spec.taskRunTemplate.podTemplate.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	string	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	string	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	string	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
tolerationSeconds	integer	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint

Property	Type	Description
		forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
<code>value</code>	<code>string</code>	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints`

Description

TopologySpreadConstraints controls how Pods are spread across your cluster among failure-domains such as regions, zones, nodes, and other user-defined topology domains.

Type

`array`

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[]`

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

`object`

Required

`maxSkew`

`topologyKey`

`whenUnsatisfiable`

Property	Type	Description
<code>labelSelector</code>	<code>object</code>	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.
<code>matchLabelKeys</code>	<code>array</code>	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).
<code>maxSkew</code>	<code>integer</code>	MaxSkew describes the degree to which pods may be unevenly distributed. When <code>whenUnsatisfiable=DoNotSchedule</code>, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone

Property	Type	Description
		cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. zone1 zone2 zone3 P P P P P - if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). - if MaxSkew is 2, incoming pod can be scheduled onto any zone. When <code>whenUnsatisfiable=ScheduleAnyway</code>, it is used to give higher precedence to topologies that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.
<code>minDomains</code>	<code>integer</code>	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule.

Property	Type	Description
		For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: zone1 zone2 zone3 P P P P P P The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.
nodeAffinityPolicy	string	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are: - Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. - Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations. If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
nodeTaintsPolicy	string	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are: Honor: nodes without taints, along with tainted nodes for which the incoming pod has a

Property	Type	Description
		toleration, are included. Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
<code>topologyKey</code>	<code>string</code>	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It's a required field.
<code>whenUnsatisfiable</code>	<code>string</code>	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. - DoNotSchedule (default) tells the scheduler not to schedule it. - ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher

Property	Type	Description
		precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: zone1 zone2 zone3 P P P P P If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it <i>more</i> imbalanced. It's a required field.

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].labelSelector`

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
<code>matchLabels</code>	<code>object</code>	<code>matchLabels</code> is a map of {key,value} pairs. A single {key,value} in the <code>matchLabels</code> map is equivalent to an element of <code>matchExpressions</code> , whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions`

Description

`matchExpressions` is a list of label selector requirements. The requirements are ANDed.

Type

`array`

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key`

`operator`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the label key that the selector applies to.
<code>operator</code>	<code>string</code>	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
<code>values</code>	<code>array</code>	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

`.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values[]`

Type

`string`

.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].matchLabelKeys

Description

MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

Type

array

.spec.taskRunTemplate.podTemplate.topologySpreadConstraints[].matchLabelKeys[]

Type

string

.spec.timeouts

Description

Time after which the Pipeline times out. Currently three keys are accepted in the map pipeline, tasks and finally with `Timeouts.pipeline >= Timeouts.tasks + Timeouts.finally`

Type

object

Property	Type	Description
<code>finally</code>	<code>string</code>	Finally sets the maximum allowed duration of this pipeline's finally
<code>pipeline</code>	<code>string</code>	Pipeline sets the maximum allowed duration for execution of the entire pipeline. The sum of individual timeouts for tasks and finally must not exceed this value.
<code>tasks</code>	<code>string</code>	Tasks sets the maximum allowed duration of this pipeline's tasks

.spec.workspaces

Description

Workspaces holds a set of workspace bindings that must match names with those declared in the pipeline.

Type

array

.spec.workspaces[]

Description

WorkspaceBinding maps a Task's declared workspace to a Volume.

Type

`object`

Required

`name`

Property	Type	Description
<code>configMap</code>	<code>object</code>	ConfigMap represents a configMap that should populate workspace.
<code>csi</code>	<code>object</code>	CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.
<code>emptyDir</code>	<code>object</code>	EmptyDir represents a temporary directory that shares Task's lifetime. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir Either this OR PersistentVolumeClaim can be used.
<code>name</code>	<code>string</code>	Name is the name of the workspace populated by the volume.
<code>persistentVolumeClaim</code>	<code>object</code>	PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.
<code>projected</code>	<code>object</code>	Projected represents a projected volume that should populate this workspace.

Property	Type	Description
<code>secret</code>	<code>object</code>	Secret represents a secret that should populate this workspace.
<code>subPath</code>	<code>string</code>	SubPath is optionally a directory on the volume which s be used for this binding (i.e. the volume will be mounter sub directory).
<code>volumeClaimTemplate</code>		VolumeClaimTemplate is a template for a claim that will created in the same namespace. The PipelineRun cont responsible for creating a unique claim for each instanc PipelineRun. See PersistentVolumeClaim (API version:

.spec.workspaces[].configMap

Description

ConfigMap represents a configMap that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	defaultMode is optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file

Property	Type	Description
		mode, like fsGroup, and the result can be other mode bits set.
items	array	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	optional specify whether the ConfigMap or its keys must be defined

.spec.workspaces[].configMap.items

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will

error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.workspaces[].configMap.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.spec.workspaces[].csi

Description

CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.

Type

object

Required

driver

Property	Type	Description
<code>driver</code>	<code>string</code>	driver is the name of the CSI driver that handles this volume. Consult with your admin for the correct name as registered in the cluster.
<code>fsType</code>	<code>string</code>	fsType to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.
<code>nodePublishSecretRef</code>	<code>object</code>	nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

Property	Type	Description
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> specifies a read-only configuration for the volume. Defaults to false (read/write).
<code>volumeAttributes</code>	<code>object</code>	<code>volumeAttributes</code> stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

`.spec.workspaces[].csi.nodePublishSecretRef`

Description

`nodePublishSecretRef` is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI `NodePublishVolume` and `NodeUnpublishVolume` calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names

`.spec.workspaces[].csi.volumeAttributes`

Description

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

Type

object

.spec.workspaces[].emptyDir

Description

EmptyDir represents a temporary directory that shares a Task's lifetime. More info: <https://kubernetes.io/docs/concepts/storage/volumes#emptydir> Either this OR PersistentVolumeClaim can be used.

Type

object

Property	Type	Description
<code>medium</code>	<code>string</code>	medium represents what type of storage medium should back this directory. The default is "" which means to use the node's default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir
<code>sizeLimit</code>		sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

`.spec.workspaces[].persistentVolumeClaim`

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.

Type

object

Required

claimName

Property	Type	Description
<code>claimName</code>	<code>string</code>	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
<code>readOnly</code>	<code>boolean</code>	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

`.spec.workspaces[].projected`

Description

Projected represents a projected volume that should populate this workspace.

Type

object

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> are the mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code>, and the result can be other mode bits set.
<code>sources</code>	<code>array</code>	<code>sources</code> is the list of volume projections. Each entry in this list handles one source.

`.spec.workspaces[].projected.sources`

Description

`sources` is the list of volume projections. Each entry in this list handles one source.

Type

`array`

`.spec.workspaces[].projected.sources[]`

Description

Projection that may be projected along with other supported volume types. Exactly one of these fields must be set.

Type

`object`

Property	Type	Description
<code>clusterTrustBundle</code>	<code>object</code>	ClusterTrustBundle allows a pod to access the <code>.spec.trustBundle</code> field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.
<code>configMap</code>	<code>object</code>	configMap information about the configMap data to project
<code>downwardAPI</code>	<code>object</code>	downwardAPI information about the downwardAPI data to project
<code>secret</code>	<code>object</code>	secret information about the secret data to project

Property	Type	Description
<code>serviceAccountToken</code>	<code>object</code>	serviceAccountToken is information about the serviceAccountToken data to project

`.spec.workspaces[].projected.sources[].clusterTrustBundle`

Description

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.`

Type

`object`

Required

`path`

Property	Type	Description
<code>labelSelector</code>	<code>object</code>	Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

Property	Type	Description
<code>name</code>	<code>string</code>	Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.
<code>optional</code>	<code>boolean</code>	If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.
<code>path</code>	<code>string</code>	Relative path from the volume root to write the bundle.
<code>signerName</code>	<code>string</code>	Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all selected ClusterTrustBundles will be unified and deduplicated.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector`

Description

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

Type

`object`

Property	Type	Description
<code>matchExpressions</code>	<code>array</code>	<code>matchExpressions</code> is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	<code>object</code>	<code>matchLabels</code> is a map of {key,value} pairs. A single {key,value} in the <code>matchLabels</code> map is equivalent to an element of <code>matchExpressions</code> , whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions`

Description

`matchExpressions` is a list of label selector requirements. The requirements are ANDed.

Type

`array`

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key`

`operator`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the label key that the selector applies to.
<code>operator</code>	<code>string</code>	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
<code>values</code>	<code>array</code>	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values[]`

Type

`string`

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchLabels`

Description

`matchLabels` is a map of {key,value} pairs. A single {key,value} in the `matchLabels` map is equivalent to an element of `matchExpressions`, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

`.spec.workspaces[].projected.sources[].configMap`

Description

`configMap` information about the `configMap` data to project

Type

object

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	optional specify whether the ConfigMap or its keys must be defined

`.spec.workspaces[].projected.sources[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].projected.sources[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key`

`path`

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.spec.workspaces[].projected.sources[].downwardAPI

Description

downwardAPI information about the downwardAPI data to project

Type

object

Property	Type	Description
items	array	Items is a list of DownwardAPIVolume file

`.spec.workspaces[].projected.sources[].downwardAPI.items`

Description

Items is a list of DownwardAPIVolume file

Type

array

`.spec.workspaces[].projected.sources[].downwardAPI.items[]`

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

object

Required

path

Property	Type	Description
<code>fieldRef</code>	<code>object</code>	Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Property	Type	Description
<code>mode</code>	<code>integer</code>	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

`.spec.workspaces[].projected.sources[].downwardAPI.items[].fieldRef`

Description

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.workspaces[].projected.sources[].downwardAPI.items[].resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"

Property	Type	Description
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.workspaces[].projected.sources[].secret`

Description

secret information about the secret data to project

Type

`object`

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its key must be defined

`.spec.workspaces[].projected.sources[].secret.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].projected.sources[].secret.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key``path`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the key to project.

Property	Type	Description
<code>mode</code>	<code>integer</code>	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.workspaces[].projected.sources[].serviceAccountToken`

Description

serviceAccountToken is information about the serviceAccountToken data to project

Type

`object`

Required

`path`

Property	Type	Description
<code>audience</code>	<code>string</code>	audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.
<code>expirationSeconds</code>	<code>integer</code>	expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token. The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.
<code>path</code>	<code>string</code>	path is the path relative to the mount point of the file to project the token into.

`.spec.workspaces[].secret`

Description

Secret represents a secret that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> is Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code>, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	<code>items</code> If unspecified, each key-value pair in the <code>Data</code> field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the <code>..</code> path or start with <code>..</code>.
<code>optional</code>	<code>boolean</code>	<code>optional</code> field specify whether the Secret or its keys must be defined
<code>secretName</code>	<code>string</code>	<code>secretName</code> is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

.spec.workspaces[].secret.items

Description

items If unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.workspaces[].secret.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

Property	Type	Description
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.status

Description

PipelineRunStatus defines the observed state of PipelineRun

Type

`object`

Property	Type	Description
<code>annotations</code>	<code>object</code>	Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.
<code>childReferences</code>	<code>array</code>	list of TaskRun and Run names, PipelineTask names, and API versions/kinds for children of this PipelineRun.
<code>completionTime</code>	<code>string</code>	CompletionTime is the time the PipelineRun completed.

Property	Type	Description
<code>conditions</code>	<code>array</code>	Conditions the latest available observations of a resource's current state.
<code>finallyStartTime</code>	<code>string</code>	FinallyStartTime is when all non-finally tasks have been completed and only finally tasks are being executed.
<code>observedGeneration</code>	<code>integer</code>	ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.
<code>pipelineSpec</code>	<code></code>	PipelineSpec contains the exact spec used to instantiate the run. See Pipeline.spec (API version: tekton.dev/v1)
<code>provenance</code>	<code>object</code>	Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).
<code>results</code>	<code>array</code>	Results are the list of results written out by the pipeline task's containers
<code>skippedTasks</code>	<code>array</code>	list of tasks that were skipped due to when expressions evaluating to false

Property	Type	Description
<code>spanContext</code>	<code>object</code>	SpanContext contains tracing span context fields
<code>startTime</code>	<code>string</code>	StartTime is the time the PipelineRun is actually started.

.status.annotations

Description

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

Type

`object`

.status.childReferences

Description

list of TaskRun and Run names, PipelineTask names, and API versions/kinds for children of this PipelineRun.

Type

`array`

.status.childReferences[]

Description

ChildStatusReference is used to point to the statuses of individual TaskRuns and Runs within this PipelineRun.

Type

`object`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	
<code>displayName</code>	<code>string</code>	DisplayName is a user-facing name of the pipelineTask that may be used to populate a UI.
<code>kind</code>	<code>string</code>	
<code>name</code>	<code>string</code>	Name is the name of the TaskRun or Run this is referencing.
<code>pipelineTaskName</code>	<code>string</code>	PipelineTaskName is the name of the PipelineTask this is referencing.
<code>whenExpressions</code>	<code>array</code>	WhenExpressions is the list of checks guarding the execution of the PipelineTask

`.status.childReferences[].whenExpressions`

Description

WhenExpressions is the list of checks guarding the execution of the PipelineTask

Type

`array`

`.status.childReferences[].whenExpressions[]`

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

object

Property	Type	Description
<code>cel</code>	<code>string</code>	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md ↗
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.status.childReferences[].whenExpressions[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

array

`.status.childReferences[].whenExpressions[].values[]`

Type

string

.status.conditions

Description

Conditions the latest available observations of a resource's current state.

Type

array

.status.conditions[]

Description

Condition defines a readiness condition for a Knative resource. See: <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties>

Type

object

Required

status

type

Property	Type	Description
<code>lastTransitionTime</code>	string	LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).

Property	Type	Description
<code>message</code>	<code>string</code>	A human readable message indicating details about the transition.
<code>reason</code>	<code>string</code>	The reason for the condition's last transition.
<code>severity</code>	<code>string</code>	Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.
<code>status</code>	<code>string</code>	Status of the condition, one of True, False, Unknown.
<code>type</code>	<code>string</code>	Type of condition.

.status.provenance

Description

Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).

Type

`object`

Property	Type	Description
<code>featureFlags</code>	<code>object</code>	FeatureFlags identifies the feature flags that were used during the task/pipeline run

Property	Type	Description
<code>refSource</code>	<code>object</code>	RefSource identifies the source where a remote task/pipeline came from.

.status.provenance.featureFlags

Description

FeatureFlags identifies the feature flags that were used during the task/pipeline run

Type

`object`

Property	Type	Description
<code>awaitSidecarReadiness</code>	<code>boolean</code>	
<code>coschedule</code>	<code>string</code>	
<code>disableCredsInit</code>	<code>boolean</code>	
<code>disableInlineSpec</code>	<code>string</code>	
<code>enableAPIFields</code>	<code>string</code>	
<code>enableArtifacts</code>	<code>boolean</code>	
<code>enableCELInWhenExpression</code>	<code>boolean</code>	
<code>enableConciseResolverSyntax</code>	<code>boolean</code>	
<code>enableKeepPodOnCancel</code>	<code>boolean</code>	
<code>enableKubernetesSidecar</code>	<code>boolean</code>	
<code>enableParamEnum</code>	<code>boolean</code>	

Property	Type	Description
<code>enableProvenanceInStatus</code>	<code>boolean</code>	
<code>enableStepActions</code>	<code>boolean</code>	EnableStepActions is a no-op flag since StepActions are stable
<code>enforceNonfalsifiability</code>	<code>string</code>	
<code>maxResultSize</code>	<code>integer</code>	
<code>requireGitSSHSecretKnownHosts</code>	<code>boolean</code>	
<code>resultExtractionMethod</code>	<code>string</code>	
<code>runningInEnvWithInjectedSidecars</code>	<code>boolean</code>	
<code>sendCloudEventsForRuns</code>	<code>boolean</code>	
<code>setSecurityContext</code>	<code>boolean</code>	
<code>setSecurityContextReadOnlyRootFilesystem</code>	<code>boolean</code>	
<code>verificationNoMatchPolicy</code>	<code>string</code>	VerificationNoMatchPolicy is the feature flag for "trusted-resources-verification-no-match-policy" VerificationNoMatchPolicy can be set to "ignore", "warn" and "fail" values. ignore: skip trusted resources verification when no matching verification policies found warn: skip trusted

Property	Type	Description
		resources verification when no matching verification policies found and log a warning fail: fail the taskrun or pipelines run if no matching verification policies found

.status.provenance.refSource

Description

RefSource identifies the source where a remote task/pipeline came from.

Type

object

Property	Type	Description
<code>digest</code>	<code>object</code>	Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}
<code>entryPoint</code>	<code>string</code>	EntryPoint identifies the entry point into the build. This is often a path to a build definition file and/or a target label within that file. Example: "task/git-clone/0.10/git-clone.yaml"
<code>uri</code>	<code>string</code>	URI indicates the identity of the source of the build definition. Example: " https://github.com/tektoncd/catalog "

.status.provenance.refSource.digest

Description

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

Type

object

.status.results

Description

Results are the list of results written out by the pipeline task's containers

Type

array

.status.results[]

Description

PipelineRunResult used to describe the results of a pipeline

Type

object

Required

name

value

Property	Type	Description
name	string	Name is the result's name as declared by the Pipeline
value		Value is the result returned from the execution of this PipelineRun

`.status.skippedTasks`

Description

list of tasks that were skipped due to when expressions evaluating to false

Type

`array`

`.status.skippedTasks[]`

Description

SkippedTask is used to describe the Tasks that were skipped due to their When Expressions evaluating to False. This is a struct because we are looking into including more details about the When Expressions that caused this Task to be skipped.

Type

`object`

Required

`name``reason`

Property	Type	Description
<code>name</code>	<code>string</code>	Name is the Pipeline Task name
<code>reason</code>	<code>string</code>	Reason is the cause of the PipelineTask being skipped.
<code>whenExpressions</code>	<code>array</code>	WhenExpressions is the list of checks guarding the execution of the PipelineTask

`.status.skippedTasks[].whenExpressions`

Description

WhenExpressions is the list of checks guarding the execution of the PipelineTask

Type

array

.status.skippedTasks[].whenExpressions[]

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

object

Property	Type	Description
<code>cel</code>	<code>string</code>	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md ↗
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.status.skippedTasks[].whenExpressions[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

array

`.status.skippedTasks[].whenExpressions[].values[]`

Type

string

`.status.spanContext`

Description

SpanContext contains tracing span context fields

Type

object

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1/namespaces/{namespace}/pipelineruns`
 - `DELETE` : delete collection of PipelineRun
 - `GET` : list objects of kind PipelineRun
 - `POST` : create a new PipelineRun
- `/apis/tekton.dev/v1/namespaces/{namespace}/pipelineruns/{name}`
 - `DELETE` : delete the specified PipelineRun

- **GET** : read the specified PipelineRun
- **PATCH** : partially update the specified PipelineRun
- **PUT** : replace the specified PipelineRun
- `/apis/tekton.dev/v1/namespaces/{namespace}/pipelineruns/{name}/status`
 - **GET** : read status of the specified PipelineRun
 - **PATCH** : partially update status of the specified PipelineRun
 - **PUT** : replace status of the specified PipelineRun

/apis/tekton.dev/v1/namespaces/{namespace}/pipelineruns

HTTP method

DELETE

Description

delete collection of PipelineRun

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind PipelineRun

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRunList</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new PipelineRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>PipelineRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRun</code> schema
201 - Created	<code>PipelineRun</code> schema
202 - Accepted	<code>PipelineRun</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1/namespaces/{namespace}/pipelineruns/{name}

HTTP method

DELETE

Description

delete the specified PipelineRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
202 - Accepted	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified PipelineRun

HTTP responses

HTTP code	Response body
200 - OK	PipelineRun schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified PipelineRun

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
fieldValidation	string	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified PipelineRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>PipelineRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRun</code> schema
201 - Created	<code>PipelineRun</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1/namespaces/{namespace}/pipelineruns/{name}/status

HTTP method

GET

Description

read status of the specified PipelineRun

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRun</code> schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified PipelineRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default

Parameter	Type	Description
		in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRun</code> schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified PipelineRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request

Parameter	Type	Description
		will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>PipelineRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>PipelineRun</code> schema
201 - Created	<code>PipelineRun</code> schema
401 - Unauthorized	Empty



TaskRun [tekton.dev/v1]

Description

TaskRun represents a single execution of a Task. TaskRuns are how the steps specified in a Task are executed; they specify the parameters and resources used to run the steps in a Task.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
<code>metadata</code>	<code>ObjectMeta</code>	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	TaskRunSpec defines the desired state of TaskRun
<code>status</code>	<code>object</code>	TaskRunStatus defines the observed state of TaskRun

.spec

Description

TaskRunSpec defines the desired state of TaskRun

Type

`object`

Property	Type	Description
<code>computeResources</code>	<code>object</code>	Compute resources to use for this TaskRun

Property	Type	Description
<code>debug</code>	<code>object</code>	TaskRunDebug defines the breakpoint config for a particular TaskRun
<code>params</code>	<code>array</code>	Params is a list of Param
<code>podTemplate</code>	<code>object</code>	PodTemplate holds pod specific configuration
<code>retries</code>	<code>integer</code>	Retries represents how many times this TaskRun should be retried in the event of task failure.
<code>serviceAccountName</code>	<code>string</code>	
<code>sidecarSpecs</code>	<code>array</code>	Specs to apply to Sidecars in this TaskRun. If a field is specified in both a Sidecar and a SidecarSpec, the value from the SidecarSpec will be used. This field is only supported when the alpha feature gate is enabled.
<code>status</code>	<code>string</code>	Used for cancelling a TaskRun (and maybe more later on)
<code>statusMessage</code>	<code>string</code>	Status message for cancellation.
<code>stepSpecs</code>	<code>array</code>	Specs to apply to Steps in this TaskRun. If a field is specified in both a Step and a StepSpec, the value

Property	Type	Description
		from the StepSpec will be used. This field is only supported when the alpha feature gate is enabled.
<code>taskRef</code>	<code>object</code>	no more than one of the TaskRef and TaskSpec may be specified.
<code>taskSpec</code>	<code>object</code>	Specifying TaskSpec can be disabled by setting <code>disable-inline-spec</code> feature flag. See Task.spec (API version: tekton.dev/v1)
<code>timeout</code>	<code>string</code>	Time after which one retry attempt times out. Defaults to 1 hour. Refer Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration
<code>workspaces</code>	<code>array</code>	Workspaces is a list of WorkspaceBindings from volumes to workspaces.

.spec.computeResources

Description

Compute resources to use for this TaskRun

Type

`object`

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
claims	array	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.computeResources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

.spec.computeResources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

.spec.computeResources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.computeResources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.debug

Description

TaskRunDebug defines the breakpoint config for a particular TaskRun

Type

object

Property	Type	Description
<code>breakpoints</code>	object	TaskBreakpoints defines the breakpoint config for a particular Task

.spec.debug.breakpoints

Description

TaskBreakpoints defines the breakpoint config for a particular Task

Type

object

Property	Type	Description
<code>beforeSteps</code>	array	

Property	Type	Description
<code>onFailure</code>	<code>string</code>	if enabled, pause TaskRun on failure of a step failed step will not exit

`.spec.debug.breakpoints.beforeSteps`

Type

`array`

`.spec.debug.breakpoints.beforeSteps[]`

Type

`string`

`.spec.params`

Description

Params is a list of Param

Type

`array`

`.spec.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

.spec.podTemplate

Description

PodTemplate holds pod specific configuration

Type

`object`

Property	Type	Description
<code>affinity</code>		If specified, the pod's scheduling constraints. Pod.spec.affinity (API version: v1)
<code>automountServiceAccountToken</code>	<code>boolean</code>	AutomountServiceAccountToken indicates whether the pod is running as this service account should have a token automatically mounted.
<code>dnsConfig</code>	<code>object</code>	Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated configuration based on DNSPolicy.
<code>dnsPolicy</code>	<code>string</code>	Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. Parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

Property	Type	Description
<code>enableServiceLinks</code>	<code>boolean</code>	<code>EnableServiceLinks</code> indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Defaults to true.
<code>env</code>	<code>array</code>	List of environment variables that can be provided to containers belonging to the pod.
<code>hostAliases</code>	<code>array</code>	<code>HostAliases</code> is an optional list of hosts and IP addresses that will be injected into the pod's hosts file if specified. Only valid for non-hostNetwork pods.
<code>hostNetwork</code>	<code>boolean</code>	<code>HostNetwork</code> specifies whether the pod may use the host's network namespace.
<code>imagePullSecrets</code>	<code>array</code>	<code>ImagePullSecrets</code> gives the name of the secret in the pod's namespace to use when pulling images. This field is optional, as it is defaulted by the pod's namespace.
<code>nodeSelector</code>	<code>object</code>	<code>NodeSelector</code> is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/node-label/

Property	Type	Description
<code>priorityClassName</code>	<code>string</code>	If specified, indicates the pod's priority. "system-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities in the namespace: "system-cluster-critical" is higher than "system-critical". Any other name can be used to specify a priority class, which will be defined by creating a PriorityClass object with the corresponding name. If not specified, the pod priority will be default, which is zero if there is no default.
<code>runtimeClassName</code>	<code>string</code>	RuntimeClassName refers to a RuntimeClass object in the node.k8s.io group, which should be used to run this pod. If no RuntimeClass resource matches the pod class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is a default class with an empty definition that uses the default runtime handler. More info: https://git.k8s.io/enhancements/keps/sig-node/runtime-class.md This is a beta feature as of Kubernetes v1.14.
<code>schedulerName</code>	<code>string</code>	SchedulerName specifies the scheduler to be used to dispatch the Pod
<code>securityContext</code>		SecurityContext holds pod-level security attributes and common container settings. Optional: Default securityContext will be set if not provided. See type description for default values of each field. Pod.spec.securityContext (API version: v1)

Property	Type	Description
<code>tolerations</code>	<code>array</code>	If specified, the pod's tolerations.
<code>topologySpreadConstraints</code>	<code>array</code>	TopologySpreadConstraints controls how Pods spread across your cluster among failure-domains as regions, zones, nodes, and other user-defined topology domains.
<code>volumes</code>		List of volumes that can be mounted by containers belonging to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes See Pod.spec.volumes (API version: v1)

.spec.podTemplate.dnsConfig

Description

Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

Type

`object`

Property	Type	Description
<code>nameservers</code>	<code>array</code>	A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Property	Type	Description
<code>options</code>	<code>array</code>	A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.
<code>searches</code>	<code>array</code>	A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

`.spec.podTemplate.dnsConfig.nameservers`

Description

A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Type

`array`

`.spec.podTemplate.dnsConfig.nameservers[]`

Type

`string`

`.spec.podTemplate.dnsConfig.options`

Description

A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.

Type

`array`

`.spec.podTemplate.dnsConfig.options[]`

Description

PodDNSConfigOption defines DNS resolver options of a pod.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name is this DNS resolver option's name. Required.
<code>value</code>	<code>string</code>	Value is this DNS resolver option's value.

`.spec.podTemplate.dnsConfig.searches`

Description

A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

Type

`array`

`.spec.podTemplate.dnsConfig.searches[]`

Type

`string`

`.spec.podTemplate.env`

Description

List of environment variables that can be provided to the containers belonging to the pod.

Type

array

.spec.podTemplate.env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.
value	string	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".
valueFrom	object	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.podTemplate.env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

object

Property	Type	Description
<code>configMapKeyRef</code>	object	Selects a key of a ConfigMap.
<code>fieldRef</code>	object	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	object	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	object	Selects a key of a secret in the pod's namespace

`.spec.podTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

object

Required

key

Property	Type	Description
key	string	The key to select.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	Specify whether the ConfigMap or its key must be defined

.spec.podTemplate.env[].valueFrom.fieldRef**Description**

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Type

object

Required

fieldPath

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.podTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.podTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
key	string	The key of the secret to select from. Must be a valid secret key.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	Specify whether the Secret or its key must be defined

`.spec.podTemplate.hostAliases`

Description

HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.

Type

array

`.spec.podTemplate.hostAliases[]`

Description

HostAlias holds the mapping between IP and hostnames that will be injected as an entry in the pod's hosts file.

Type

`object`

Required

`ip`

Property	Type	Description
<code>hostnames</code>	<code>array</code>	Hostnames for the above IP address.
<code>ip</code>	<code>string</code>	IP address of the host file entry.

`.spec.podTemplate.hostAliases[].hostnames`

Description

Hostnames for the above IP address.

Type

`array`

`.spec.podTemplate.hostAliases[].hostnames[]`

Type

`string`

`.spec.podTemplate.imagePullSecrets`

Description

ImagePullSecrets gives the name of the secret used by the pod to pull the image if specified

Type

array

.spec.podTemplate.imagePullSecrets[]

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.podTemplate.nodeSelector

Description

NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info: <https://kubernetes.io/docs/concepts/configuration/assign-pod-node/>

Type

object

.spec.podTemplate.tolerations

Description

If specified, the pod's tolerations.

Type

array

.spec.podTemplate.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	string	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	string	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	string	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so

Property	Type	Description
		that a pod can tolerate all taints of a particular category.
<code>tolerationSeconds</code>	<code>integer</code>	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
<code>value</code>	<code>string</code>	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

`.spec.podTemplate.topologySpreadConstraints`

Description

TopologySpreadConstraints controls how Pods are spread across your cluster among failure-domains such as regions, zones, nodes, and other user-defined topology domains.

Type

`array`

`.spec.podTemplate.topologySpreadConstraints[]`

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

`object`

Required

`maxSkew``topologyKey``whenUnsatisfiable`

Property	Type	Description
<code>labelSelector</code>	<code>object</code>	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.
<code>matchLabelKeys</code>	<code>array</code>	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).
<code>maxSkew</code>	<code>integer</code>	MaxSkew describes the degree to which pods may be unevenly distributed. When <code>whenUnsatisfiable=DoNotSchedule</code>, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the

Property	Type	Description
		minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. zone1 zone2 zone3 P P P P P - if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). - if MaxSkew is 2, incoming pod can be scheduled onto any zone. When <code>whenUnsatisfiable=ScheduleAnyway</code>, it is used to give higher precedence to topologies that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.
<code>minDomains</code>	<code>integer</code>	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid

Property	Type	Description
		values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule. For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: zone1 zone2 zone3 P P P P P P The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.
nodeAffinityPolicy	string	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are: - Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. - Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations. If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
nodeTaintsPolicy	string	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are:

Property	Type	Description
		- Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included. - Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
<code>topologyKey</code>	<code>string</code>	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It's a required field.
<code>whenUnsatisfiable</code>	<code>string</code>	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. DoNotSchedule (default) tells the scheduler not to schedule it.

Property	Type	Description
		ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: zone1 zone2 zone3 P P P P P If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it <i>more</i> imbalanced. It's a required field.

`.spec.podTemplate.topologySpreadConstraints[].labelSelector`

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
<code>matchLabels</code>	<code>object</code>	<code>matchLabels</code> is a map of {key,value} pairs. A single {key,value} in the <code>matchLabels</code> map is equivalent to an element of <code>matchExpressions</code> , whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions`

Description

`matchExpressions` is a list of label selector requirements. The requirements are ANDed.

Type

`array`

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

`object`

Required

`key`

`operator`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the label key that the selector applies to.
<code>operator</code>	<code>string</code>	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
<code>values</code>	<code>array</code>	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

`array`

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values[]`

Type

`string`

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchLabels`

Description

`matchLabels` is a map of {key,value} pairs. A single {key,value} in the `matchLabels` map is equivalent to an element of `matchExpressions`, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

`.spec.podTemplate.topologySpreadConstraints[].matchLabelKeys`

Description

`MatchLabelKeys` is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with `labelSelector` to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both `MatchLabelKeys` and `LabelSelector`. `MatchLabelKeys` cannot be set when `LabelSelector` isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against `labelSelector`. This is a beta field and requires the `MatchLabelKeysInPodTopologySpread` feature gate to be enabled (enabled by default).

Type

array

`.spec.podTemplate.topologySpreadConstraints[].matchLabelKeys[]`

Type

string

`.spec.sidecarSpecs`

Description

Specs to apply to Sidecars in this TaskRun. If a field is specified in both a Sidecar and a SidecarSpec, the value from the SidecarSpec will be used. This field is only supported when the alpha feature gate is enabled.

Type

array

.spec.sidecarSpecs[]

Description

TaskRunSidecarSpec is used to override the values of a Sidecar in the corresponding Task.

Type

object

Required

computeResources

name

Property	Type	Description
<code>computeResources</code>	<code>object</code>	The resource requirements to apply to the Sidecar.
<code>name</code>	<code>string</code>	The name of the Sidecar to override.

.spec.sidecarSpecs[].computeResources

Description

The resource requirements to apply to the Sidecar.

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
claims	array	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.sidecarSpecs[].computeResources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

`.spec.sidecarSpecs[].computeResources.claims[]`

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.spec.sidecarSpecs[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.sidecarSpecs[].computeResources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.stepSpecs

Description

Specs to apply to Steps in this TaskRun. If a field is specified in both a Step and a StepSpec, the value from the StepSpec will be used. This field is only supported when the alpha feature gate is enabled.

Type

array

.spec.stepSpecs[]

Description

TaskRunStepSpec is used to override the values of a Step in the corresponding Task.

Type

object

Required

computeResources

name

Property	Type	Description
<code>computeResources</code>	object	The resource requirements to apply to the Step.
<code>name</code>	string	The name of the Step to override.

`.spec.stepSpecs[].computeResources`

Description

The resource requirements to apply to the Step.

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in <code>spec.resourceClaims</code> , that are used by this container.
<code>claims</code>	<code>array</code>	This is an alpha field and requires enabling the <code>DynamicResourceAllocation</code> feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.stepSpecs[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.spec.stepSpecs[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.spec.stepSpecs[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.stepSpecs[].computeResources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.taskRef

Description

no more than one of the TaskRef and TaskSpec may be specified.

Type

object

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task
<code>kind</code>	<code>string</code>	TaskKind indicates the Kind of the Task: - Namespaced Task when Kind is set to "Task". If Kind is "", it defaults to "Task". - Custom Task when Kind is non-empty and APIVersion is non-empty

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names ↗
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.taskRef.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.taskRef.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

name

value

Property	Type	Description
name	string	
value		

.spec.workspaces

Description

Workspaces is a list of WorkspaceBindings from volumes to workspaces.

Type

array

.spec.workspaces[]

Description

WorkspaceBinding maps a Task's declared workspace to a Volume.

Type

object

Required

name

Property	Type	Description
configMap	object	ConfigMap represents a configMap that should populate workspace.
csi	object	CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.

Property	Type	Description
<code>emptyDir</code>	<code>object</code>	EmptyDir represents a temporary directory that shares Task's lifetime. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir Either this OR PersistentVolumeClaim can be used.
<code>name</code>	<code>string</code>	Name is the name of the workspace populated by the volume
<code>persistentVolumeClaim</code>	<code>object</code>	PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.
<code>projected</code>	<code>object</code>	Projected represents a projected volume that should populate this workspace.
<code>secret</code>	<code>object</code>	Secret represents a secret that should populate this workspace.
<code>subPath</code>	<code>string</code>	SubPath is optionally a directory on the volume which should be used for this binding (i.e. the volume will be mounted to the sub directory).
<code>volumeClaimTemplate</code>	<code>object</code>	VolumeClaimTemplate is a template for a claim that will be created in the same namespace. The PipelineRun controller is responsible for creating a unique claim for each instance of PipelineRun. See PersistentVolumeClaim (API version: v1)

.spec.workspaces[].configMap

Description

ConfigMap represents a configMap that should populate this workspace.

Type

object

Property	Type	Description
defaultMode	integer	defaultMode is optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
items	array	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	optional specify whether the ConfigMap or its keys must be defined

`.spec.workspaces[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key`

`path`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the key to project.
<code>mode</code>	<code>integer</code>	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.workspaces[].csi`

Description

CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.

Type

`object`

Required

`driver`

Property	Type	Description
<code>driver</code>	<code>string</code>	driver is the name of the CSI driver that handles this volume. Consult with your admin for the

Property	Type	Description
		correct name as registered in the cluster.
<code>fsType</code>	<code>string</code>	<code>fsType</code> to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.
<code>nodePublishSecretRef</code>	<code>object</code>	<code>nodePublishSecretRef</code> is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> specifies a read-only configuration for the volume. Defaults to false (read/write).
<code>volumeAttributes</code>	<code>object</code>	<code>volumeAttributes</code> stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

`.spec.workspaces[].csi.nodePublishSecretRef`

Description

`nodePublishSecretRef` is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume

calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.workspaces[].csi.volumeAttributes

Description

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

Type

object

.spec.workspaces[].emptyDir

Description

EmptyDir represents a temporary directory that shares a Task's lifetime. More info: <https://kubernetes.io/docs/concepts/storage/volumes#emptydir> Either this OR PersistentVolumeClaim can be used.

Type

object

Property	Type	Description
<code>medium</code>	<code>string</code>	medium represents what type of storage medium should back this directory. The default is "" which means to use the node's default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir
<code>sizeLimit</code>		sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

`.spec.workspaces[].persistentVolumeClaim`

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.

Type

`object`

Required

`claimName`

Property	Type	Description
<code>claimName</code>	<code>string</code>	<code>claimName</code> is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> Will force the ReadOnly setting in VolumeMounts. Default false.

`.spec.workspaces[].projected`

Description

Projected represents a projected volume that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> are the mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code> , and the result can be other mode bits set.

Property	Type	Description
<code>sources</code>	<code>array</code>	<code>sources</code> is the list of volume projections. Each entry in this list handles one source.

`.spec.workspaces[].projected.sources`

Description

`sources` is the list of volume projections. Each entry in this list handles one source.

Type

`array`

`.spec.workspaces[].projected.sources[]`

Description

Projection that may be projected along with other supported volume types. Exactly one of these fields must be set.

Type

`object`

Property	Type	Description
<code>clusterTrustBundle</code>	<code>object</code>	ClusterTrustBundle allows a pod to access the <code>.spec.trustBundle</code> field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector.

Property	Type	Description
		Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.
<code>configMap</code>	<code>object</code>	configMap information about the configMap data to project
<code>downwardAPI</code>	<code>object</code>	downwardAPI information about the downwardAPI data to project
<code>secret</code>	<code>object</code>	secret information about the secret data to project
<code>serviceAccountToken</code>	<code>object</code>	serviceAccountToken is information about the serviceAccountToken data to project

`.spec.workspaces[].projected.sources[].clusterTrustBundle`

Description

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem.`

Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.

Type

object

Required

path

Property	Type	Description
labelSelector	object	Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".
name	string	Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.
optional	boolean	If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.
path	string	Relative path from the volume root to write the bundle.
signerName	string	Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all

Property	Type	Description
		selected ClusterTrustBundles will be unified and deduplicated.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector`

Description

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	object	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions`

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

key

operator

Property	Type	Description
key	string	key is the label key that the selector applies to.
operator	string	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	array	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

array

.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values[]

Type

string

.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

.spec.workspaces[].projected.sources[].configMap

Description

configMap information about the configMap data to project

Type

object

Property	Type	Description
items	array	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
optional	boolean	optional specify whether the ConfigMap or its keys must be defined

`.spec.workspaces[].projected.sources[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

`.spec.workspaces[].projected.sources[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.workspaces[].projected.sources[].downwardAPI`

Description

downwardAPI information about the downwardAPI data to project

Type

object

Property	Type	Description
items	array	Items is a list of DownwardAPIVolume file

`.spec.workspaces[].projected.sources[].downwardAPI.items`

Description

Items is a list of DownwardAPIVolume file

Type

array

`.spec.workspaces[].projected.sources[].downwardAPI.items[]`

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

object

Required

path

Property	Type	Description
<code>fieldRef</code>	<code>object</code>	Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.
<code>mode</code>	<code>integer</code>	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

`.spec.workspaces[].projected.sources[].downwardAPI.items[].fieldRef`

Description

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Type

object

Required

fieldPath

Property	Type	Description
apiVersion	string	Version of the schema the FieldPath is written in terms of, defaults to "v1".
fieldPath	string	Path of the field to select in the specified API version.

.spec.workspaces[].projected.sources[].downwardAPI.items[].resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

object

Required

resource

Property	Type	Description
containerName	string	Container name: required for volumes, optional for env vars

Property	Type	Description
<code>divisor</code>	<code>integer</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.workspaces[].projected.sources[].secret`

Description

secret information about the secret data to project

Type

`object`

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its key must be defined

`.spec.workspaces[].projected.sources[].secret.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].projected.sources[].secret.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key`

`path`

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.spec.workspaces[].projected.sources[].serviceAccountToken

Description

serviceAccountToken is information about the serviceAccountToken data to project

Type

object

Required

path

Property	Type	Description
<code>audience</code>	<code>string</code>	audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.
<code>expirationSeconds</code>	<code>integer</code>	expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token. The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.
<code>path</code>	<code>string</code>	path is the path relative to the mount point of the file to project the token into.

`.spec.workspaces[].secret`

Description

Secret represents a secret that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> is Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code>, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	<code>items</code> If unspecified, each key-value pair in the <code>Data</code> field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the <code>..</code> path or start with <code>..</code>.
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its keys must be defined
<code>secretName</code>	<code>string</code>	<code>secretName</code> is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

.spec.workspaces[].secret.items

Description

items If unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.workspaces[].secret.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

Property	Type	Description
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.status

Description

TaskRunStatus defines the observed state of TaskRun

Type

`object`

Required

`podName`

Property	Type	Description
<code>annotations</code>	<code>object</code>	Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.
<code>artifacts</code>	<code>object</code>	Artifacts are the list of artifacts written out by the task's containers
<code>completionTime</code>	<code>string</code>	CompletionTime is the time the build completed.

Property	Type	Description
<code>conditions</code>	<code>array</code>	Conditions the latest available observations of a resource's current state.
<code>observedGeneration</code>	<code>integer</code>	ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.
<code>podName</code>	<code>string</code>	PodName is the name of the pod responsible for executing this task's steps.
<code>provenance</code>	<code>object</code>	Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).
<code>results</code>	<code>array</code>	Results are the list of results written out by the task's containers
<code>retriesStatus</code>	<code></code>	RetriesStatus contains the history of TaskRunStatus in case of a retry in order to keep record of failures. All TaskRunStatus stored in RetriesStatus will have no date within the RetriesStatus as is redundant.
<code>sidecars</code>	<code>array</code>	The list has one entry per sidecar in the manifest. Each entry is represents the imageid of the corresponding sidecar.

Property	Type	Description
<code>spanContext</code>	<code>object</code>	SpanContext contains tracing span context fields
<code>startTime</code>	<code>string</code>	StartTime is the time the build is actually started.
<code>steps</code>	<code>array</code>	Steps describes the state of each build step container.
<code>taskSpec</code>	<code>object</code>	TaskSpec contains the Spec from the dereferenced Task definition used to instantiate this TaskRun.

.status.annotations

Description

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

Type

`object`

.status.artifacts

Description

Artifacts are the list of artifacts written out by the task's containers

Type

`object`

Property	Type	Description
<code>inputs</code>	<code>array</code>	
<code>outputs</code>	<code>array</code>	

`.status.artifacts.inputs`

Type

`array`

`.status.artifacts.inputs[]`

Description

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

Type

`object`

Property	Type	Description
<code>buildOutput</code>	<code>boolean</code>	Indicate if the artifact is a build output or a by-product
<code>name</code>	<code>string</code>	The artifact's identifying category name
<code>values</code>	<code>array</code>	A collection of values related to the artifact

`.status.artifacts.inputs[].values`

Description

A collection of values related to the artifact

Type

array

.status.artifacts.inputs[].values[]

Description

ArtifactValue represents a specific value or data element within an Artifact.

Type

object

Property	Type	Description
digest	object	
uri	string	

.status.artifacts.inputs[].values[].digest

Type

object

.status.artifacts.outputs

Type

array

.status.artifacts.outputs[]

Description

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

Type

`object`

Property	Type	Description
<code>buildOutput</code>	<code>boolean</code>	Indicate if the artifact is a build output or a by-product
<code>name</code>	<code>string</code>	The artifact's identifying category name
<code>values</code>	<code>array</code>	A collection of values related to the artifact

`.status.artifacts.outputs[].values`

Description

A collection of values related to the artifact

Type

`array`

`.status.artifacts.outputs[].values[]`

Description

ArtifactValue represents a specific value or data element within an Artifact.

Type

`object`

Property	Type	Description
<code>digest</code>	<code>object</code>	
<code>uri</code>	<code>string</code>	

`.status.artifacts.outputs[].values[].digest`

Type

`object`

`.status.conditions`

Description

Conditions the latest available observations of a resource's current state.

Type

`array`

`.status.conditions[]`

Description

Condition defines a readiness condition for a Knative resource. See: <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties>

Type

`object`

Required

`status``type`

Property	Type	Description
<code>lastTransitionTime</code>	<code>string</code>	LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).

Property	Type	Description
<code>message</code>	<code>string</code>	A human readable message indicating details about the transition.
<code>reason</code>	<code>string</code>	The reason for the condition's last transition.
<code>severity</code>	<code>string</code>	Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.
<code>status</code>	<code>string</code>	Status of the condition, one of True, False, Unknown.
<code>type</code>	<code>string</code>	Type of condition.

.status.provenance

Description

Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).

Type

`object`

Property	Type	Description
<code>featureFlags</code>	<code>object</code>	FeatureFlags identifies the feature flags that were used during the task/pipeline run

Property	Type	Description
<code>refSource</code>	<code>object</code>	RefSource identifies the source where a remote task/pipeline came from.

`.status.provenance.featureFlags`

Description

FeatureFlags identifies the feature flags that were used during the task/pipeline run

Type

`object`

Property	Type	Description
<code>awaitSidecarReadiness</code>	<code>boolean</code>	
<code>coschedule</code>	<code>string</code>	
<code>disableCredsInit</code>	<code>boolean</code>	
<code>disableInlineSpec</code>	<code>string</code>	
<code>enableAPIFields</code>	<code>string</code>	
<code>enableArtifacts</code>	<code>boolean</code>	
<code>enableCELInWhenExpression</code>	<code>boolean</code>	
<code>enableConciseResolverSyntax</code>	<code>boolean</code>	
<code>enableKeepPodOnCancel</code>	<code>boolean</code>	
<code>enableKubernetesSidecar</code>	<code>boolean</code>	
<code>enableParamEnum</code>	<code>boolean</code>	

Property	Type	Description
<code>enableProvenanceInStatus</code>	<code>boolean</code>	
<code>enableStepActions</code>	<code>boolean</code>	EnableStepActions is a no-op flag since StepActions are stable
<code>enforceNonfalsifiability</code>	<code>string</code>	
<code>maxResultSize</code>	<code>integer</code>	
<code>requireGitSSHSecretKnownHosts</code>	<code>boolean</code>	
<code>resultExtractionMethod</code>	<code>string</code>	
<code>runningInEnvWithInjectedSidecars</code>	<code>boolean</code>	
<code>sendCloudEventsForRuns</code>	<code>boolean</code>	
<code>setSecurityContext</code>	<code>boolean</code>	
<code>setSecurityContextReadOnlyRootFilesystem</code>	<code>boolean</code>	
<code>verificationNoMatchPolicy</code>	<code>string</code>	VerificationNoMatchPolicy is the feature flag for "trusted-resources-verification-no-match-policy" VerificationNoMatchPolicy can be set to "ignore", "warn" and "fail" values. ignore: skip trusted resources verification when no matching verification policies found warn: skip trusted

Property	Type	Description
		resources verification when no matching verification policies found and log a warning fail: fail the taskrun or pipelines run if no matching verification policies found

.status.provenance.refSource

Description

RefSource identifies the source where a remote task/pipeline came from.

Type

object

Property	Type	Description
<code>digest</code>	<code>object</code>	Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}
<code>entryPoint</code>	<code>string</code>	EntryPoint identifies the entry point into the build. This is often a path to a build definition file and/or a target label within that file. Example: "task/git-clone/0.10/git-clone.yaml"
<code>uri</code>	<code>string</code>	URI indicates the identity of the source of the build definition. Example: " https://github.com/tektoncd/catalog "

.status.provenance.refSource.digest

Description

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

Type

object

.status.results

Description

Results are the list of results written out by the task's containers

Type

array

.status.results[]

Description

TaskRunResult used to describe the results of a task

Type

object

Required

name

value

Property	Type	Description
name	string	Name the given name
type	string	Type is the user-specified type of the result. The possible type is currently "string" and will support "array" in following work.

Property	Type	Description
<code>value</code>	<code>string</code>	Value the given value of the result

`.status.sidecars`

Description

The list has one entry per sidecar in the manifest. Each entry is represents the imageid of the corresponding sidecar.

Type

`array`

`.status.sidecars[]`

Description

SidecarState reports the results of running a sidecar in a Task.

Type

`object`

Property	Type	Description
<code>container</code>	<code>string</code>	
<code>imageID</code>	<code>string</code>	
<code>name</code>	<code>string</code>	
<code>running</code>	<code>object</code>	Details about a running container
<code>terminated</code>	<code>object</code>	Details about a terminated container

Property	Type	Description
<code>waiting</code>	<code>object</code>	Details about a waiting container

`.status.sidecars[].running`

Description

Details about a running container

Type

`object`

Property	Type	Description
<code>startedAt</code>	<code>string</code>	Time at which the container was last (re-)started

`.status.sidecars[].terminated`

Description

Details about a terminated container

Type

`object`

Required

`exitCode`

Property	Type	Description
<code>containerID</code>	<code>string</code>	Container's ID in the format <code>'!//<container_id>'</code>

Property	Type	Description
<code>exitCode</code>	<code>integer</code>	Exit status from the last termination of the container
<code>finishedAt</code>	<code>string</code>	Time at which the container last terminated
<code>message</code>	<code>string</code>	Message regarding the last termination of the container
<code>reason</code>	<code>string</code>	(brief) reason from the last termination of the container
<code>signal</code>	<code>integer</code>	Signal from the last termination of the container
<code>startedAt</code>	<code>string</code>	Time at which previous execution of the container started

`.status.sidecars[].waiting`

Description

Details about a waiting container

Type

`object`

Property	Type	Description
<code>message</code>	<code>string</code>	Message regarding why the container is not yet running.

Property	Type	Description
<code>reason</code>	<code>string</code>	(brief) reason the container is not yet running.

`.status.spanContext`

Description

SpanContext contains tracing span context fields

Type

`object`

`.status.steps`

Description

Steps describes the state of each build step container.

Type

`array`

`.status.steps[]`

Description

StepState reports the results of running a step in a Task.

Type

`object`

Property	Type	Description
<code>container</code>	<code>string</code>	
<code>imageID</code>	<code>string</code>	

Property	Type	Description
<code>inputs</code>	<code>array</code>	
<code>name</code>	<code>string</code>	
<code>outputs</code>	<code>array</code>	
<code>provenance</code>	<code>object</code>	Provenance contains metadata about resources used in the TaskRun/PipelineRun such as the source from where a remote build definition was fetched. This field aims to carry minimum amount of metadata in *Run status so that Tekton Chains can capture them in the provenance.
<code>results</code>	<code>array</code>	
<code>running</code>	<code>object</code>	Details about a running container
<code>terminated</code>	<code>object</code>	Details about a terminated container
<code>terminationReason</code>	<code>string</code>	
<code>waiting</code>	<code>object</code>	Details about a waiting container

`.status.steps[].inputs`

Type

`array`

`.status.steps[].inputs[]`

Description

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

Type

object

Property	Type	Description
<code>buildOutput</code>	<code>boolean</code>	Indicate if the artifact is a build output or a by-product
<code>name</code>	<code>string</code>	The artifact's identifying category name
<code>values</code>	<code>array</code>	A collection of values related to the artifact

`.status.steps[].inputs[].values`

Description

A collection of values related to the artifact

Type

array

`.status.steps[].inputs[].values[]`

Description

ArtifactValue represents a specific value or data element within an Artifact.

Type

object

Property	Type	Description
<code>digest</code>	<code>object</code>	
<code>uri</code>	<code>string</code>	

`.status.steps[].inputs[].values[].digest`

Type

`object`

`.status.steps[].outputs`

Type

`array`

`.status.steps[].outputs[]`

Description

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

Type

`object`

Property	Type	Description
<code>buildOutput</code>	<code>boolean</code>	Indicate if the artifact is a build output or a by-product
<code>name</code>	<code>string</code>	The artifact's identifying category name

Property	Type	Description
<code>values</code>	<code>array</code>	A collection of values related to the artifact

`.status.steps[].outputs[].values`

Description

A collection of values related to the artifact

Type

`array`

`.status.steps[].outputs[].values[]`

Description

ArtifactValue represents a specific value or data element within an Artifact.

Type

`object`

Property	Type	Description
<code>digest</code>	<code>object</code>	
<code>uri</code>	<code>string</code>	

`.status.steps[].outputs[].values[].digest`

Type

`object`

`.status.steps[].provenance`

Description

Provenance contains metadata about resources used in the TaskRun/PipelineRun such as the source from where a remote build definition was fetched. This field aims to carry minimum amount of metadata in *Run status so that Tekton Chains can capture them in the provenance.

Type

object

Property	Type	Description
<code>featureFlags</code>	object	FeatureFlags identifies the feature flags that were used during the task/pipeline run
<code>refSource</code>	object	RefSource identifies the source where a remote task/pipeline came from.

.status.steps[].provenance.featureFlags

Description

FeatureFlags identifies the feature flags that were used during the task/pipeline run

Type

object

Property	Type	Description
<code>awaitSidecarReadiness</code>	boolean	
<code>coschedule</code>	string	
<code>disableCredsInit</code>	boolean	
<code>disableInlineSpec</code>	string	

Property	Type	Description
<code>enableAPIFields</code>	<code>string</code>	
<code>enableArtifacts</code>	<code>boolean</code>	
<code>enableCELInWhenExpression</code>	<code>boolean</code>	
<code>enableConciseResolverSyntax</code>	<code>boolean</code>	
<code>enableKeepPodOnCancel</code>	<code>boolean</code>	
<code>enableKubernetesSidecar</code>	<code>boolean</code>	
<code>enableParamEnum</code>	<code>boolean</code>	
<code>enableProvenanceInStatus</code>	<code>boolean</code>	
<code>enableStepActions</code>	<code>boolean</code>	EnableStepActions is a no-op flag since StepActions are stable
<code>enforceNonfalsifiability</code>	<code>string</code>	
<code>maxResultSize</code>	<code>integer</code>	
<code>requireGitSSHSecretKnownHosts</code>	<code>boolean</code>	
<code>resultExtractionMethod</code>	<code>string</code>	
<code>runningInEnvWithInjectedSidecars</code>	<code>boolean</code>	
<code>sendCloudEventsForRuns</code>	<code>boolean</code>	
<code>setSecurityContext</code>	<code>boolean</code>	
<code>setSecurityContextReadOnlyRootFilesystem</code>	<code>boolean</code>	

Property	Type	Description
<code>verificationNoMatchPolicy</code>	<code>string</code>	VerificationNoMatchPolicy is the feature flag for "trusted-resources-verification-no-match-policy" VerificationNoMatchPolicy can be set to "ignore", "warn" and "fail" values. ignore: skip trusted resources verification when no matching verification policies found warn: skip trusted resources verification when no matching verification policies found and log a warning fail: fail the taskrun or pipelines run if no matching verification policies found

`.status.steps[].provenance.refSource`

Description

RefSource identifies the source where a remote task/pipeline came from.

Type

`object`

Property	Type	Description
<code>digest</code>	<code>object</code>	Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}
<code>entryPoint</code>	<code>string</code>	EntryPoint identifies the entry point into the build. This is often a path to a build definition file and/or a target label within that file. Example: "task/git-clone/0.10/git-clone.yaml"
<code>uri</code>	<code>string</code>	URI indicates the identity of the source of the build definition. Example: " https://github.com/tektoncd/catalog "

`.status.steps[].provenance.refSource.digest`

Description

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

Type

`object`

`.status.steps[].results`

Type

`array`

`.status.steps[].results[]`

Description

TaskRunResult used to describe the results of a task

Type

object

Required

name

value

Property	Type	Description
name	string	Name the given name
type	string	Type is the user-specified type of the result. The possible type is currently "string" and will support "array" in following work.
value		Value the given value of the result

.status.steps[].running

Description

Details about a running container

Type

object

Property	Type	Description
startedAt	string	Time at which the container was last (re-)started

.status.steps[].terminated

Description

Details about a terminated container

Type

`object`

Required

`exitCode`

Property	Type	Description
<code>containerID</code>	<code>string</code>	Container's ID in the format '://<container_id>'
<code>exitCode</code>	<code>integer</code>	Exit status from the last termination of the container
<code>finishedAt</code>	<code>string</code>	Time at which the container last terminated
<code>message</code>	<code>string</code>	Message regarding the last termination of the container
<code>reason</code>	<code>string</code>	(brief) reason from the last termination of the container
<code>signal</code>	<code>integer</code>	Signal from the last termination of the container
<code>startedAt</code>	<code>string</code>	Time at which previous execution of the container started

`.status.steps[].waiting`

Description

Details about a waiting container

Type

`object`

Property	Type	Description
<code>message</code>	<code>string</code>	Message regarding why the container is not yet running.
<code>reason</code>	<code>string</code>	(brief) reason the container is not yet running.

.status.taskSpec

Description

TaskSpec contains the Spec from the dereferenced Task definition used to instantiate this TaskRun.

Type

`object`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a user-facing description of the task that may be used to populate a UI.
<code>displayName</code>	<code>string</code>	DisplayName is a user-facing name of the task that may be used to populate a UI.

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of input parameters required to run the task. Params must be supplied as inputs in TaskRuns unless they declare a default value.
<code>results</code>	<code>array</code>	Results are values that this Task can output
<code>sidecars</code>	<code>array</code>	Sidecars are run alongside the Task's step containers. They begin before the steps start and end after the steps complete.
<code>stepTemplate</code>	<code>object</code>	StepTemplate can be used as the basis for all step containers within the Task, so that the steps inherit settings on the base container.
<code>steps</code>	<code>array</code>	Steps are the steps of the build; each step is run sequentially with the source mounted into /workspace.
<code>volumes</code>	<code></code>	Volumes is a collection of volumes that are available to mount into the steps of the build. See Pod.spec.volumes (API version: v1)
<code>workspaces</code>	<code>array</code>	Workspaces are the volumes that this Task requires.

`.status.taskSpec.params`

Description

Params is a list of input parameters required to run the task. Params must be supplied as inputs in TaskRuns unless they declare a default value.

Type

array

.status.taskSpec.params[]

Description

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources). Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

Type

object

Required

name

Property	Type	Description
<code>default</code>		Default is the value a parameter takes if no input value is supplied. If default is set, a Task may be executed without a supplied value for the parameter.
<code>description</code>	<code>string</code>	Description is a user-facing description of the parameter that may be used to populate a UI.
<code>enum</code>	<code>array</code>	Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Property	Type	Description
<code>name</code>	<code>string</code>	Name declares the name by which a parameter is referenced.
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs parameter.
<code>type</code>	<code>string</code>	Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

`.status.taskSpec.params[].enum`

Description

Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Type

`array`

`.status.taskSpec.params[].enum[]`

Type

`string`

`.status.taskSpec.params[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs parameter.

Type

object

.status.taskSpec.results

Description

Results are values that this Task can output

Type

array

.status.taskSpec.results[]

Description

TaskResult used to describe the results of a task

Type

object

Required

name

Property	Type	Description
description	string	Description is a human-readable description of the result
name	string	Name the given name
properties	object	Properties is the JSON Schema properties to support key-value pairs results.

Property	Type	Description
<code>type</code>	<code>string</code>	Type is the user-specified type of the result. The possible type is currently "string" and will support "array" in following work.
<code>value</code>		Value the expression used to retrieve the value of the result from an underlying Step.

`.status.taskSpec.results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type

`object`

`.status.taskSpec.sidecars`

Description

Sidecars are run alongside the Task's step containers. They begin before the steps start and end after the steps complete.

Type

`array`

`.status.taskSpec.sidecars[]`

Description

Sidecar has nearly the same data structure as Step but does not have the ability to timeout.

Type

`object`

Required

name		
Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell

Property	Type	Description
<code>computeResources</code>	<code>object</code>	ComputeResources required by this Sidecar. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/resources-containers/
<code>env</code>	<code>array</code>	List of environment variables to set in the Sidecar. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the Sidecar. The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the Sidecar is starting. When a key exists in multiple sources, the last associated with the last source will take precedence. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/envvars/#sources
<code>image</code>	<code>string</code>	Image reference name. More info: https://kubernetes.io/docs/concepts/containers/images/
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images/

Property	Type	Description
lifecycle	object	Actions that the management system should take Sidecar lifecycle events. Cannot be updated.
livenessProbe	object	Periodic probe of Sidecar liveness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes
name	string	Name of the Sidecar specified as a DNS_LABEL. in a Task must have a unique name (DNS_LABEL). Cannot be updated.
ports	array	List of ports to expose from the Sidecar. Exposing ports gives the system additional information about the connections a container uses, but is primarily informational. Specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default address inside a container will be accessible from the host. Cannot be updated.
readinessProbe	object	Periodic probe of Sidecar service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes

Property	Type	Description
<code>restartPolicy</code>	<code>string</code>	RestartPolicy refers to kubernetes RestartPolicy. It is set for an initContainer and must have it's policy set. It is currently left optional to help support Kubernetes prior to 1.29 when this feature was introduced.
<code>script</code>	<code>string</code>	Script is the contents of an executable file to execute. If Script is not empty, the Step cannot have an Command or Args.
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Sidecar should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>startupProbe</code>	<code>object</code>	StartupProbe indicates that the Pod the Sidecar is successfully initialized. If specified, no other probe is allowed until this completes successfully. If this probe fails, the Sidecar will be restarted, just as if the livenessProbe failed. This is useful to provide different probe parameters at the beginning of the lifecycle, when it might take a long time to load data into the cache, than during steady-state operation. This can be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes ↗
<code>stdin</code>	<code>boolean</code>	Whether this Sidecar should allocate a buffer for stdin during container runtime. If this is not set, reads from stdin will fail.

Property	Type	Description
		Sidecar will always result in EOF. Default is false.
<code>stdinOnce</code>	<code>boolean</code>	Whether the container runtime should close the stdin stream after it has been opened by a single attach. When the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on container start, is empty until the first client attaches to stdin, remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the container is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false
<code>terminationMessagePath</code>	<code>string</code>	Optional: Path at which the file to which the Sidecar message will be written is mounted into the Sidecar container. Message written is intended to be brief final status assertion failure message. Will be truncated by the container if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.
<code>terminationMessagePolicy</code>	<code>string</code>	Indicate how the termination message should be populated. File will use the contents of terminationMessagePath to populate the Sidecar status message on both success and failure. FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container exited with an error. The log output is limited to 2048 bytes or 20 lines, whichever is smaller. Defaults to File. Cannot be updated.

Property	Type	Description
<code>tty</code>	<code>boolean</code>	Whether this Sidecar should allocate a TTY for itself. Requires 'stdin' to be true. Default is false.
<code>volumeDevices</code>	<code>array</code>	volumeDevices is the list of block devices to be used by the Sidecar.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Sidecar's filesystem. Cannot be updated.
<code>workingDir</code>	<code>string</code>	Sidecar's working directory. If not specified, the container runtime's default will be used, which might be configured for a container image. Cannot be updated.
<code>workspaces</code>	<code>array</code>	This is an alpha field. You must set the "enable-alpha" flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Sidecar wants exclusive access to. Adding a workspace to the list means that any other Step or Sidecar that does not request this Workspace will not have access to it.

`.status.taskSpec.sidecars[].args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are

reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.status.taskSpec.sidecars[].args[]

Type

string

.status.taskSpec.sidecars[].command

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.status.taskSpec.sidecars[].command[]

Type

string

.status.taskSpec.sidecars[].computeResources

Description

ComputeResources required by this Sidecar. Cannot be updated. More info:
<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
<code>claims</code>	<code>array</code>	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.status.taskSpec.sidecars[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.status.taskSpec.sidecars[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.status.taskSpec.sidecars[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.status.taskSpec.sidecars[].computeResources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:

<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.status.taskSpec.sidecars[].env

Description

List of environment variables to set in the Sidecar. Cannot be updated.

Type

array

.status.taskSpec.sidecars[].env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.

Property	Type	Description
<code>value</code>	<code>string</code>	Variable references <code>\$(VAR_NAME)</code> are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to <code>""</code>.
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.status.taskSpec.sidecars[].env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code>, <code>metadata.namespace</code>, <code>metadata.labels['<KEY>']</code>, <code>metadata.annotations['<KEY>']</code>, <code>spec.nodeName</code>,

Property	Type	Description
		spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.status.taskSpec.sidecars[].env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of

Property	Type	Description
		this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.status.taskSpec.sidecars[].env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.status.taskSpec.sidecars[].env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.status.taskSpec.sidecars[].env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.status.taskSpec.sidecars[].envFrom`

Description

List of sources to populate environment variables in the Sidecar. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the container is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

`.status.taskSpec.sidecars[].envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.status.taskSpec.sidecars[].envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

.status.taskSpec.sidecars[].envFrom[].secretRef

Description

The Secret to select from

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	Specify whether the Secret must be defined

.status.taskSpec.sidecars[].lifecycle

Description

Actions that the management system should take in response to Sidecar lifecycle events.
Cannot be updated.

Type

object

Property	Type	Description
postStart	object	PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the

Property	Type	Description
		container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗
preStop	object	PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

.status.taskSpec.sidecars[].lifecycle.postStart

Description

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
exec	object	Exec specifies a command to execute in the container.

Property	Type	Description
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to perform.
<code>sleep</code>	<code>object</code>	Sleep represents a duration that the container should sleep.
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

`.status.taskSpec.sidecars[].lifecycle.postStart.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.status.taskSpec.sidecars[].lifecycle.postStart.exec.comm` and

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.status.taskSpec.sidecars[].lifecycle.postStart.exec.comm` and[]

Type

`string`

`.status.taskSpec.sidecars[].lifecycle.postStart.httpGet`

Description

HTTPGet specifies an HTTP GET request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

Property	Type	Description
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.status.taskSpec.sidecars[].lifecycle.postStart.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.status.taskSpec.sidecars[].lifecycle.postStart.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.status.taskSpec.sidecars[].lifecycle.postStart.sleep`

Description

Sleep represents a duration that the container should sleep.

Type

`object`

Required

`seconds`

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.status.taskSpec.sidecars[].lifecycle.postStart.tcpSocket`

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it

is specified.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.status.taskSpec.sidecars[].lifecycle.preStop

Description

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies a command to execute in the container.
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to perform.
<code>sleep</code>	<code>object</code>	Sleep represents a duration that the container should sleep.
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it is specified.

`.status.taskSpec.sidecars[].lifecycle.preStop.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.status.taskSpec.sidecars[].lifecycle.preStop.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.status.taskSpec.sidecars[].lifecycle.preStop.exec.command[]`

Type

`string`

`.status.taskSpec.sidecars[].lifecycle.preStop.httpGet`

Description

HTTPGet specifies an HTTP GET request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

Property	Type	Description
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.status.taskSpec.sidecars[].lifecycle.preStop.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.status.taskSpec.sidecars[].lifecycle.preStop.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.status.taskSpec.sidecars[].lifecycle.preStop.sleep`

Description

Sleep represents a duration that the container should sleep.

Type

`object`

Required

`seconds`

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.status.taskSpec.sidecars[].lifecycle.preStop.tcpSocket`

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for backward compatibility. There is no validation of this field and lifecycle hooks will fail at runtime when it

is specified.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.status.taskSpec.sidecars[].livenessProbe

Description

Periodic probe of Sidecar liveness. Container will be restarted if the probe fails. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

object

Property	Type	Description
exec	object	Exec specifies a command to execute in the
failureThreshold	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. D

Property	Type	Description
		Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies a GRPC HealthCheckRequest
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies a connection to a TCP
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It must be longer than the expected cleanup time

Property	Type	Description
		process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. The value zero indicates stop immediately via the Kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
timeoutSeconds	integer	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes

`.status.taskSpec.sidecars[].livenessProbe.exec`

Description

Exec specifies a command to execute in the container.

Type

object

Property	Type	Description
command	array	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.status.taskSpec.sidecars[].livenessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.status.taskSpec.sidecars[].livenessProbe.exec.command[ ]`

Type

`string`

`.status.taskSpec.sidecars[].livenessProbe.grpc`

Description

GRPC specifies a GRPC HealthCheckRequest.

Type

`object`

Required

`port`

Property	Type	Description
<code>port</code>	<code>integer</code>	Port number of the gRPC service. Number must be in the range 1 to 65535.

Property	Type	Description
<code>service</code>	<code>string</code>	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

`.status.taskSpec.sidecars[].livenessProbe.httpGet`

Description

HTTPGet specifies an HTTP GET request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.

Property	Type	Description
<code>port</code>	<code>integer</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.status.taskSpec.sidecars[].livenessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.status.taskSpec.sidecars[].livenessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.status.taskSpec.sidecars[].livenessProbe.tcpSocket`

Description

TCPSocket specifies a connection to a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.status.taskSpec.sidecars[].ports`

Description

List of ports to expose from the Sidecar. Exposing a port here gives the system additional information about the network connections a container uses, but is primarily informational. Not specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default "0.0.0.0" address inside a container will be accessible from the network. Cannot be updated.

Type

array

.status.taskSpec.sidecars[].ports[]

Description

ContainerPort represents a network port in a single container.

Type

object

Required

containerPort

Property	Type	Description
<code>containerPort</code>	<code>integer</code>	Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.
<code>hostIP</code>	<code>string</code>	What host IP to bind the external port to.
<code>hostPort</code>	<code>integer</code>	Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

Property	Type	Description
<code>name</code>	<code>string</code>	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.
<code>protocol</code>	<code>string</code>	Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

`.status.taskSpec.sidecars[].readinessProbe`

Description

Periodic probe of Sidecar service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

`object`

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies a command to execute in the
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies a GRPC HealthCheckRequest to send to the service.

Property	Type	Description
<code>httpGet</code>	<code>object</code>	HTTPGet specifies an HTTP GET request to
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has s before liveness probes are initiated. More in https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies a connection to a TCP
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs gracefully upon probe failure. The grace period in seconds after the processes running in the pod are sent a termination signal and the time until the processes are forcibly halted with a kill signal. Set to zero if the processes are expected to cleanup. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the

Property	Type	Description
		(no opportunity to shut down). This is a beta feature that requires enabling ProbeTerminationGracePeriodSeconds. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
timeoutSeconds	integer	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.status.taskSpec.sidecars[].readinessProbe.exec`

Description

Exec specifies a command to execute in the container.

Type

object

Property	Type	Description
command	array	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.status.taskSpec.sidecars[].readinessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

`.status.taskSpec.sidecars[].readinessProbe.exec.command[]`

Type

string

`.status.taskSpec.sidecars[].readinessProbe.grpc`

Description

GRPC specifies a GRPC HealthCheckRequest.

Type

object

Required

port

Property	Type	Description
<code>port</code>	<code>integer</code>	Port number of the gRPC service. Number must be in the range 1 to 65535.
<code>service</code>	<code>string</code>	Service is the name of the service to place in the gRPC HealthCheckRequest (see

Property	Type	Description
		https://github.com/grpc/grpc/blob/master/doc/health-checking.md ). If this is not specified, the default behavior is defined by gRPC.

.status.taskSpec.sidecars[].readinessProbe.httpGet

Description

HTTPGet specifies an HTTP GET request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

Property	Type	Description
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.status.taskSpec.sidecars[].readinessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.status.taskSpec.sidecars[].readinessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.status.taskSpec.sidecars[].readinessProbe.tcpSocket`

Description

TCPSocket specifies a connection to a TCP port.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.status.taskSpec.sidecars[].securityContext`

Description

SecurityContext defines the security options the Sidecar should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

object

Property	Type	Description
allowPrivilegeEscalation	boolean	AllowPrivilegeEscalation controls whether a process can gain more privileges than its

Property	Type	Description
		parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. <code>AllowPrivilegeEscalation</code> is true always when the container is: - run as Privileged - has <code>CAP_SYS_ADMIN</code> Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>appArmorProfile</code>	<code>object</code>	<code>appArmorProfile</code> is the AppArmor options to use by this container. If set, this profile overrides the pod's <code>appArmorProfile</code>. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>procMount</code>	<code>string</code>	<code>procMount</code> denotes the type of proc mount to use for the containers. The default value is

Property	Type	Description
		Default which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.status.taskSpec.sidecars[].securityContext.appArmorProfile`

Description

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.status.taskSpec.sidecars[].securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

.status.taskSpec.sidecars[].securityContext.capabilities.add

Description

Added capabilities

Type

array

.status.taskSpec.sidecars[].securityContext.capabilities.add[]

Description

Capability represent POSIX capabilities type

Type

string

.status.taskSpec.sidecars[].securityContext.capabilities.drop

Description

Removed capabilities

Type

array

.status.taskSpec.sidecars[].securityContext.capabilities.drop[]

Description

Capability represent POSIX capabilities type

Type

`string`

`.status.taskSpec.sidecars[].securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.status.taskSpec.sidecars[].securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

Type

object

Required

type

Property	Type	Description
localhostProfile	string	localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
type	string	type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.status.taskSpec.sidecars[].securityContext.windowOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the

value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
<code>gmsaCredentialSpec</code>	<code>string</code>	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
<code>gmsaCredentialSpecName</code>	<code>string</code>	GMSACredentialSpecName is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the

Property	Type	Description
		value specified in SecurityContext takes precedence.

`.status.taskSpec.sidecars[].startupProbe`

Description

StartupProbe indicates that the Pod the Sidecar is running in has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies a command to execute in the
<code>failureThreshold</code>	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	object	GRPC specifies a GRPC HealthCheckRequest to
<code>httpGet</code>	object	HTTPGet specifies an HTTP GET request to

Property	Type	Description
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCP socket specifies a connection to a TCP endpoint.

Property	Type	Description
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs gracefully upon probe failure. The grace period duration in seconds after the processes running in the pod are sent a termination signal and the timer begins. The processes are forcibly halted with a kill signal when a probe fails. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. If this value is nil, the pod's spec.terminationGracePeriodSeconds is used.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/pods/probe-lifecycle#container-probes

`.status.taskSpec.sidecars[].startupProbe.exec`

Description

Exec specifies a command to execute in the container.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.status.taskSpec.sidecars[].startupProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.status.taskSpec.sidecars[].startupProbe.exec.command[]`

Type

`string`

`.status.taskSpec.sidecars[].startupProbe.grpc`

Description

GRPC specifies a GRPC HealthCheckRequest.

Type

`object`

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.status.taskSpec.sidecars[].startupProbe.httpGet

Description

HTTPGet specifies an HTTP GET request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

Property	Type	Description
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.status.taskSpec.sidecars[].startupProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.status.taskSpec.sidecars[].startupProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.status.taskSpec.sidecars[].startupProbe.tcpSocket`

Description

TCPSocket specifies a connection to a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.status.taskSpec.sidecars[].volumeDevices`

Description

volumeDevices is the list of block devices to be used by the Sidecar.

Type

array

`.status.taskSpec.sidecars[].volumeDevices[]`

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

object

Required

devicePath

name

Property	Type	Description
<code>devicePath</code>	<code>string</code>	devicePath is the path inside of the container that the device will be mapped to.
<code>name</code>	<code>string</code>	name must match the name of a persistentVolumeClaim in the pod

`.status.taskSpec.sidecars[].volumeMounts`

Description

Volumes to mount into the Sidecar's filesystem. Cannot be updated.

Type

array

`.status.taskSpec.sidecars[].volumeMounts[]`

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'. <code>mountPropagation</code> determines how mounts are propagated from the host to container and the other way around. When not set, <code>MountPropagationNone</code> is used. This field is beta in 1.10. When <code>RecursiveReadOnly</code> is set to <code>IfPossible</code> or to <code>Enabled</code> , <code>MountPropagation</code> must be <code>None</code> or unspecified (which defaults to <code>None</code>).
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>recursiveReadOnly</code>	<code>string</code>	<code>RecursiveReadOnly</code> specifies whether read-only mounts should be handled recursively.

Property	Type	Description
		If ReadOnly is false, this field has no meaning and must be unspecified. If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

`.status.taskSpec.sidecars[].workspaces`

Description

This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Sidecar wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

Type

`array`

`.status.taskSpec.sidecars[].workspaces[]`

Description

WorkspaceUsage is used by a Step or Sidecar to declare that it wants isolated access to a Workspace defined in a Task.

Type

`object`

Required

`mountPath``name`

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath is the path that the workspace should be mounted to inside the Step or Sidecar, overriding any MountPath specified in the Task's WorkspaceDeclaration.
<code>name</code>	<code>string</code>	Name is the name of the workspace this Step or Sidecar wants access to.

`.status.taskSpec.stepTemplate`

Description

StepTemplate can be used as the basis for all step containers within the Task, so that the steps inherit settings on the base container.

Type

object

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell

Property	Type	Description
<code>computeResources</code>	<code>object</code>	ComputeResources required by this Step. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>env</code>	<code>array</code>	List of environment variables to set in the Step. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.
<code>image</code>	<code>string</code>	Image reference name. More info: https://kubernetes.io/docs/concepts/containers/images ↗
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Default to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#update-images ↗
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Step should run with. If set, the fields of SecurityContext override the

Property	Type	Description
		equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>volumeDevices</code>	<code>array</code>	<code>volumeDevices</code> is the list of block devices to be used by the Step.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Step's filesystem. Cannot be upda
<code>workingDir</code>	<code>string</code>	Step's working directory. If not specified, the container runtime default will be used, which might be configured in the contain image. Cannot be updated.

`.status.taskSpec.stepTemplate.args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

`array`

`.status.taskSpec.stepTemplate.args[]`

Type

string

.status.taskSpec.stepTemplate.command

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.status.taskSpec.stepTemplate.command[]

Type

string

.status.taskSpec.stepTemplate.computeResources

Description

ComputeResources required by this Step. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
claims	array	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.status.taskSpec.stepTemplate.computeResources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

`.status.taskSpec.stepTemplate.computeResources.claims`



Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.status.taskSpec.stepTemplate.computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.status.taskSpec.stepTemplate.computeResources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

`object`

`.status.taskSpec.stepTemplate.env`

Description

List of environment variables to set in the Step. Cannot be updated.

Type

`array`

`.status.taskSpec.stepTemplate.env[]`

Description

EnvVar represents an environment variable present in a Container.

Type

`object`

Required

`name`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the environment variable. Must be a C_IDENTIFIER.

Property	Type	Description
<code>value</code>	<code>string</code>	Variable references <code>\$(VAR_NAME)</code> are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to <code>""</code>.
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.status.taskSpec.stepTemplate.env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code>, <code>metadata.namespace</code>, <code>metadata.labels['<KEY>']</code>, <code>metadata.annotations['<KEY>']</code>, <code>spec.nodeName</code>,

Property	Type	Description
		spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.status.taskSpec.stepTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of

Property	Type	Description
		this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.status.taskSpec.stepTemplate.env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.status.taskSpec.stepTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.status.taskSpec.stepTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.status.taskSpec.stepTemplate.envFrom`

Description

List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

`.status.taskSpec.stepTemplate.envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.status.taskSpec.stepTemplate.envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.status.taskSpec.stepTemplate.envFrom[].secretRef`

Description

The Secret to select from

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

`.status.taskSpec.stepTemplate.securityContext`

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info:
<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

object

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its

Property	Type	Description
		parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. <code>AllowPrivilegeEscalation</code> is true always when the container is: - run as Privileged - has <code>CAP_SYS_ADMIN</code> Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>appArmorProfile</code>	<code>object</code>	<code>appArmorProfile</code> is the AppArmor options to use by this container. If set, this profile overrides the pod's <code>appArmorProfile</code>. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>procMount</code>	<code>string</code>	<code>procMount</code> denotes the type of proc mount to use for the containers. The default value is

Property	Type	Description
		Default which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.status.taskSpec.stepTemplate.securityContext.appArmorProfile`

Description

`appArmorProfile` is the AppArmor options to use by this container. If set, this profile overrides the pod's `appArmorProfile`. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.status.taskSpec.stepTemplate.securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

.status.taskSpec.stepTemplate.securityContext.capabilities.add

Description

Added capabilities

Type

array

.status.taskSpec.stepTemplate.securityContext.capabilities.add[]

Description

Capability represent POSIX capabilities type

Type

string

.status.taskSpec.stepTemplate.securityContext.capabilities.drop

Description

Removed capabilities

Type

array

.status.taskSpec.stepTemplate.securityContext.capabilities.drop[]

Description

Capability represent POSIX capabilities type

Type

`string`

`.status.taskSpec.stepTemplate.securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.status.taskSpec.stepTemplate.securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

Type

object

Required

type

Property	Type	Description
localhostProfile	string	localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
type	string	type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.status.taskSpec.stepTemplate.securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the

value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
<code>gmsaCredentialSpec</code>	<code>string</code>	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
<code>gmsaCredentialSpecName</code>	<code>string</code>	GMSACredentialSpecName is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the

Property	Type	Description
		value specified in SecurityContext takes precedence.

`.status.taskSpec.stepTemplate.volumeDevices`

Description

volumeDevices is the list of block devices to be used by the Step.

Type

array

`.status.taskSpec.stepTemplate.volumeDevices[]`

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

object

Required

devicePath

name

Property	Type	Description
<code>devicePath</code>	<code>string</code>	devicePath is the path inside of the container that the device will be mapped to.
<code>name</code>	<code>string</code>	name must match the name of a persistentVolumeClaim in the pod

`.status.taskSpec.stepTemplate.volumeMounts`

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type

array

`.status.taskSpec.stepTemplate.volumeMounts[]`

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'.
<code>mountPropagation</code>	<code>string</code>	mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10. When RecursiveReadOnly is set to IfPossible or to Enabled, MountPropagation must be None or unspecified (which defaults to None).

Property	Type	Description
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>recursiveReadOnly</code>	<code>string</code>	RecursiveReadOnly specifies whether read-only mounts should be handled recursively. If ReadOnly is false, this field has no meaning and must be unspecified. If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.

Property	Type	Description
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

`.status.taskSpec.steps`

Description

Steps are the steps of the build; each step is run sequentially with the source mounted into /workspace.

Type

`array`

`.status.taskSpec.steps[]`

Description

Step runs a subcomponent of a Task

Type

`object`

Required

`name`

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
computeResources	object	ComputeResources required by this Step. Cannot be updated. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/

Property	Type	Description
<code>env</code>	<code>array</code>	List of environment variables to set in the Step. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the Step The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.
<code>image</code>	<code>string</code>	OCI image name. More info: https://kubernetes.io/docs/concepts/containers/images ↗
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Default to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images#updating-images ↗
<code>name</code>	<code>string</code>	Name of the Step specified as a DNS_LABEL. Each Step in a Task must have a unique name.
<code>onError</code>	<code>string</code>	OnError defines the exiting behavior of a container on error can be set to [continue stopAndFail]

Property	Type	Description
<code>params</code>	<code>array</code>	Params declares parameters passed to this step action.
<code>ref</code>	<code>object</code>	Contains the reference to an existing StepAction.
<code>results</code>	<code>array</code>	Results declares StepResults produced by the Step. It can be used in an inlined Step when used to store Results <code>\$(step.results.resultName.path)</code>. It cannot be used when referencing StepActions using <code>[v1.Step.Ref]</code>. The Results declared by the StepActions will be stored here instead.
<code>script</code>	<code>string</code>	Script is the contents of an executable file to execute. If Script is not empty, the Step cannot have an Command and the Args will be passed to the Script.
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Step should run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>stderrConfig</code>	<code>object</code>	Stores configuration for the stderr stream of the step.
<code>stdoutConfig</code>	<code>object</code>	Stores configuration for the stdout stream of the step.

Property	Type	Description
<code>timeout</code>	<code>string</code>	Timeout is the time after which the step times out. Defaults to never. Refer to Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration ↗
<code>volumeDevices</code>	<code>array</code>	<code>volumeDevices</code> is the list of block devices to be used by the Step.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Step's filesystem. Cannot be updated.
<code>when</code>	<code>array</code>	When is a list of when expressions that need to be true for the task to run
<code>workingDir</code>	<code>string</code>	Step's working directory. If not specified, the container runtime default will be used, which might be configured in the container image. Cannot be updated.
<code>workspaces</code>	<code>array</code>	This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Step wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

`.status.taskSpec.steps[].args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.status.taskSpec.steps[].args[]`

Type

string

`.status.taskSpec.steps[].command`

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.status.taskSpec.steps[].command[]`

Type

`string`

`.status.taskSpec.steps[].computeResources`

Description

ComputeResources required by this Step. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/configuration/management-resources-containers/>

Type

`object`

Property	Type	Description
		Claims lists the names of resources, defined in <code>spec.resourceClaims</code> , that are used by this container.
<code>claims</code>	<code>array</code>	This is an alpha field and requires enabling the <code>DynamicResourceAllocation</code> feature gate. This field is immutable. It can only be set for containers.
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/management-resources-containers/
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/management-resources-containers/

`.status.taskSpec.steps[].computeResources.claims`

Description

Claims lists the names of resources, defined in `spec.resourceClaims`, that are used by this container. This is an alpha field and requires enabling the `DynamicResourceAllocation` feature gate. This field is immutable. It can only be set for containers.

Type

array

`.status.taskSpec.steps[].computeResources.claims[]`

Description

ResourceClaim references one entry in `PodSpec.ResourceClaims`.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name must match the name of one entry in <code>pod.spec.resourceClaims</code> of the Pod where this field is used. It makes that resource available inside a container.
<code>request</code>	<code>string</code>	Request is the name chosen for a request in the referenced claim. If empty, everything from the claim is made available, otherwise only the result of this request.

`.status.taskSpec.steps[].computeResources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info:
<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.status.taskSpec.steps[].computeResources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info:
<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.status.taskSpec.steps[].env

Description

List of environment variables to set in the Step. Cannot be updated.

Type

array

.status.taskSpec.steps[].env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the environment variable. Must be a C_IDENTIFIER.
<code>value</code>	<code>string</code>	Variable references <code>\$(VAR_NAME)</code> are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code>, which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code>. Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to <code>""</code>.
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.status.taskSpec.steps[].env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.

Property	Type	Description
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code> , <code>metadata.namespace</code> , <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , <code>spec.nodeName</code> , <code>spec.serviceAccountName</code> , <code>status.hostIP</code> , <code>status.podIP</code> , <code>status.podIPs</code> .
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (<code>limits.cpu</code> , <code>limits.memory</code> , <code>limits.ephemeral-storage</code> , <code>requests.cpu</code> , <code>requests.memory</code> and <code>requests.ephemeral-storage</code>) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.status.taskSpec.steps[].env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.status.taskSpec.steps[].env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".

Property	Type	Description
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.status.taskSpec.steps[].env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.status.taskSpec.steps[].env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
key	string	The key of the secret to select from. Must be a valid secret key.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	Specify whether the Secret or its key must be defined

.status.taskSpec.steps[].envFrom

Description

List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the Step is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

array

`.status.taskSpec.steps[].envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

object

Property	Type	Description
<code>configMapRef</code>	object	The ConfigMap to select from
<code>prefix</code>	string	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	object	The Secret to select from

`.status.taskSpec.steps[].envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

object

Property	Type	Description
<code>name</code>	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.status.taskSpec.steps[].envFrom[].secretRef`

Description

The Secret to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

`.status.taskSpec.steps[].params`

Description

Params declares parameters passed to this step action.

Type

array

.status.taskSpec.steps[].params[]

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

.status.taskSpec.steps[].ref

Description

Contains the reference to an existing StepAction.

Type

object

Property	Type	Description
name	string	Name of the referenced step

Property	Type	Description
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.status.taskSpec.steps[].ref.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.status.taskSpec.steps[].ref.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	

Property	Type	Description
<code>value</code>	<code>object</code>	

`.status.taskSpec.steps[].results`

Description

Results declares StepResults produced by the Step. It can be used in an inlined Step when used to store Results to `$(step.results.resultName.path)`. It cannot be used when referencing StepActions using `[v1.Step.Ref]`. The Results declared by the StepActions will be stored here instead.

Type

`array`

`.status.taskSpec.steps[].results[]`

Description

StepResult used to describe the Results of a Step.

Type

`object`

Required

`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a human-readable description of the result
<code>name</code>	<code>string</code>	Name the given name

Property	Type	Description
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs results.
<code>type</code>	<code>string</code>	The possible types are 'string', 'array', and 'object', with 'string' as the default.

`.status.taskSpec.steps[].results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type

`object`

`.status.taskSpec.steps[].securityContext`

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info:

<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

`object`

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the

Property	Type	Description
		container process. AllowPrivilegeEscalation is true always when the container is: - run as Privileged - has CAP_SYS_ADMIN Note that this field cannot be set when spec.os.name is windows.
<code>appArmorProfile</code>	<code>object</code>	appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when spec.os.name is windows.
<code>procMount</code>	<code>string</code>	procMount denotes the type of proc mount to use for the containers. The default value is Default which uses the container runtime defaults for readonly paths and masked

Property	Type	Description
		paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

Property	Type	Description
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and

Property	Type	Description
		PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.status.taskSpec.steps[].securityContext.appArmorProfile`

Description

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

Type

object

Required

type

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.status.taskSpec.steps[].securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

`.status.taskSpec.steps[].securityContext.capabilities.add`

Description

Added capabilities

Type

`array`

`.status.taskSpec.steps[].securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

`string`

`.status.taskSpec.steps[].securityContext.capabilities.drop`

Description

Removed capabilities

Type

array

`.status.taskSpec.steps[].securityContext.capabilities.drop`

Description

Capability represent POSIX capabilities type

Type

string

`.status.taskSpec.steps[].securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.

Property	Type	Description
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.status.taskSpec.steps[].securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when `spec.os.name` is `windows`.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	<code>localhostProfile</code> indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
<code>type</code>	<code>string</code>	<code>type</code> indicates which kind of seccomp profile will be applied. Valid options are:

Property	Type	Description
		Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.status.taskSpec.steps[].securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
gmsaCredentialSpec	string	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
gmsaCredentialSpecName	string	GMSACredentialSpecName is the name of the GMSA credential spec to use.
hostProcess	boolean	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective

Property	Type	Description
		HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

`.status.taskSpec.steps[].stderrConfig`

Description

Stores configuration for the stderr stream of the step.

Type

`object`

Property	Type	Description
<code>path</code>	<code>string</code>	Path to duplicate stdout stream to on container's local filesystem.

`.status.taskSpec.steps[].stdoutConfig`

Description

Stores configuration for the stdout stream of the step.

Type

object

Property	Type	Description
path	string	Path to duplicate stdout stream to on container's local filesystem.

.status.taskSpec.steps[].volumeDevices

Description

volumeDevices is the list of block devices to be used by the Step.

Type

array

.status.taskSpec.steps[].volumeDevices[]

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

object

Required

devicePath

name

Property	Type	Description
devicePath	string	devicePath is the path inside of the container that the device will be mapped to.
name		

Property	Type	Description
<code>name</code>	<code>string</code>	name must match the name of a persistentVolumeClaim in the pod

`.status.taskSpec.steps[].volumeMounts`

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type

`array`

`.status.taskSpec.steps[].volumeMounts[]`

Description

VolumeMount describes a mounting of a Volume within a container.

Type

`object`

Required

`mountPath` `name`

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'.
<code>mountPropagation</code>	<code>string</code>	mountPropagation determines how mounts are propagated from the host to container and the other way around. When not set, MountPropagationNone is used. This field is beta in 1.10. When

Property	Type	Description
		RecursiveReadOnly is set to IfPossible or to Enabled, MountPropagation must be None or unspecified (which defaults to None).
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>recursiveReadOnly</code>	<code>string</code>	RecursiveReadOnly specifies whether read-only mounts should be handled recursively. If ReadOnly is false, this field has no meaning and must be unspecified. If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.

Property	Type	Description
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

`.status.taskSpec.steps[].when`

Description

When is a list of when expressions that need to be true for the task to run

Type

`array`

`.status.taskSpec.steps[].when[]`

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

`object`

Property	Type	Description
<code>cel</code>	<code>string</code>	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md ↗
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.status.taskSpec.steps[].when[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

`array`

`.status.taskSpec.steps[].when[].values[]`

Type

`string`

`.status.taskSpec.steps[].workspaces`

Description

This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Step wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

Type

`array`

`.status.taskSpec.steps[].workspaces[]`

Description

WorkspaceUsage is used by a Step or Sidecar to declare that it wants isolated access to a Workspace defined in a Task.

Type

`object`

Required

`mountPath``name`

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath is the path that the workspace should be mounted to inside the Step or Sidecar, overriding any MountPath specified in the Task's WorkspaceDeclaration.
<code>name</code>	<code>string</code>	Name is the name of the workspace this Step or Sidecar wants access to.

`.status.taskSpec.workspaces`

Description

Workspaces are the volumes that this Task requires.

Type

array

.status.taskSpec.workspaces[]

Description

WorkspaceDeclaration is a declaration of a volume that a Task requires.

Type

object

Required

name

Property	Type	Description
<code>description</code>	<code>string</code>	Description is an optional human readable description of this volume.
<code>mountPath</code>	<code>string</code>	MountPath overrides the directory that the volume will be made available at.
<code>name</code>	<code>string</code>	Name is the name by which you can bind the volume at runtime.
<code>optional</code>	<code>boolean</code>	Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.

Property	Type	Description
<code>readOnly</code>	<code>boolean</code>	ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1/namespaces/{namespace}/taskruns`
 - `DELETE` : delete collection of TaskRun
 - `GET` : list objects of kind TaskRun
 - `POST` : create a new TaskRun
- `/apis/tekton.dev/v1/namespaces/{namespace}/taskruns/{name}`
 - `DELETE` : delete the specified TaskRun
 - `GET` : read the specified TaskRun
 - `PATCH` : partially update the specified TaskRun
 - `PUT` : replace the specified TaskRun
- `/apis/tekton.dev/v1/namespaces/{namespace}/taskruns/{name}/status`
 - `GET` : read status of the specified TaskRun
 - `PATCH` : partially update status of the specified TaskRun
 - `PUT` : replace status of the specified TaskRun

`/apis/tekton.dev/v1/namespaces/{namespace}/taskruns`

HTTP method

`DELETE`

Description

delete collection of TaskRun

HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind TaskRun

HTTP responses

HTTP code	Response body
200 - OK	TaskRunList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new TaskRun

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+. - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>TaskRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code> schema
201 - Created	<code>TaskRun</code> schema
202 - Accepted	<code>TaskRun</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1/namespaces/{namespace}/taskruns/{name}

HTTP method

DELETE

Description

delete the specified TaskRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code>  schema
202 - Accepted	<code>Status</code>  schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified TaskRun

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code>  schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified TaskRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified TaskRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>TaskRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code> schema
201 - Created	<code>TaskRun</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1/namespaces/{namespace}/taskruns/{name}/status

HTTP method

`GET`

Description

read status of the specified TaskRun

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified TaskRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified `TaskRun`

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>TaskRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>TaskRun</code> schema
201 - Created	<code>TaskRun</code> schema
401 - Unauthorized	Empty



ClusterTask [tekton.dev/v1]

Description

ClusterTask is a Task with a cluster scope. ClusterTasks are used to represent Tasks that should be publicly addressable from any namespace in the cluster. Deprecated: Please use the cluster resolver instead.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

Property	Type	Description
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
<code>metadata</code>	<code>ObjectMeta</code>	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the desired state of the Task from the client

.spec

Description

Spec holds the desired state of the Task from the client

Type

`object`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a user-facing description of the task that may be used to populate a UI.
<code>displayName</code>	<code>string</code>	DisplayName is a user-facing name of the task that may be used to populate a UI.

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of input parameters required to run the task. Params must be supplied as inputs in TaskRuns unless they declare a default value.
<code>resources</code>	<code>object</code>	Resources is a list input and output resource to run the task Resources are represented in TaskRuns as bindings to instances of PipelineResources. Deprecated: Unused, preserved only for backwards compatibility
<code>results</code>	<code>array</code>	Results are values that this Task can output
<code>sidecars</code>	<code>array</code>	Sidecars are run alongside the Task's step containers. They begin before the steps start and end after the steps complete.
<code>stepTemplate</code>	<code>object</code>	StepTemplate can be used as the basis for all step containers within the Task, so that the steps inherit settings on the base container.
<code>steps</code>	<code>array</code>	Steps are the steps of the build; each step is run sequentially with the source mounted into /workspace.

Property	Type	Description
<code>volumes</code>	<code>array</code>	Volumes is a collection of volumes that are available to mount into the steps of the build.
<code>workspaces</code>	<code>array</code>	Workspaces are the volumes that this Task requires.

`.spec.params`

Description

Params is a list of input parameters required to run the task. Params must be supplied as inputs in TaskRuns unless they declare a default value.

Type

`array`

`.spec.params[]`

Description

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources). Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

Type

`object`

Required

`name`

Property	Type	Description
<code>default</code>	<code>object</code>	Default is the value a parameter takes if no input value is supplied. If default is set, a Task may be executed without a supplied value for the parameter.
<code>description</code>	<code>string</code>	Description is a user-facing description of the parameter that may be used to populate a UI.
<code>enum</code>	<code>array</code>	Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.
<code>name</code>	<code>string</code>	Name declares the name by which a parameter is referenced.
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs parameter.
<code>type</code>	<code>string</code>	Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

`.spec.params[].default`

Description

Default is the value a parameter takes if no input value is supplied. If default is set, a Task may be executed without a supplied value for the parameter.

Type

object

.spec.params[].enum

Description

Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Type

array

.spec.params[].enum[]

Type

string

.spec.params[].properties

Description

Properties is the JSON Schema properties to support key-value pairs parameter.

Type

object

.spec.resources

Description

Resources is a list input and output resource to run the task Resources are represented in TaskRuns as bindings to instances of PipelineResources. Deprecated: Unused, preserved only for backwards compatibility

Type

object

Property	Type	Description
<code>inputs</code>	<code>array</code>	Inputs holds the mapping from the PipelineResources declared in DeclaredPipelineResources to the input PipelineResources required by the Task.
<code>outputs</code>	<code>array</code>	Outputs holds the mapping from the PipelineResources declared in DeclaredPipelineResources to the input PipelineResources required by the Task.

`.spec.resources.inputs`

Description

Inputs holds the mapping from the PipelineResources declared in DeclaredPipelineResources to the input PipelineResources required by the Task.

Type

`array`

`.spec.resources.inputs[]`

Description

TaskResource defines an input or output Resource declared as a requirement by a Task. The Name field will be used to refer to these Resources within the Task definition, and when provided as an Input, the Name will be the path to the volume mounted containing this Resource as an input (e.g. an input Resource named ``workspace`` will be mounted at ``/workspace``). Deprecated: Unused, preserved only for backwards compatibility

Type

`object`

Required

`name`

`type`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a user-facing description of the declared resource that may be used to populate a UI.
<code>name</code>	<code>string</code>	Name declares the name by which a resource is referenced in the definition. Resources may be referenced by name in the definition of a Task's steps.
<code>optional</code>	<code>boolean</code>	Optional declares the resource as optional. By default optional is set to false which makes a resource required. optional: true - the resource is considered optional optional: false - the resource is considered required (equivalent of not specifying it)
<code>targetPath</code>	<code>string</code>	TargetPath is the path in workspace directory where the resource will be copied.
<code>type</code>	<code>string</code>	Type is the type of this resource;

`.spec.resources.outputs`

Description

Outputs holds the mapping from the PipelineResources declared in DeclaredPipelineResources to the input PipelineResources required by the Task.

Type

`array`

.spec.resources.outputs[]

Description

TaskResource defines an input or output Resource declared as a requirement by a Task. The Name field will be used to refer to these Resources within the Task definition, and when provided as an Input, the Name will be the path to the volume mounted containing this Resource as an input (e.g. an input Resource named `workspace` will be mounted at `/workspace`). Deprecated: Unused, preserved only for backwards compatibility

Type

object

Required

name

type

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a user-facing description of the declared resource that may be used to populate a UI.
<code>name</code>	<code>string</code>	Name declares the name by which a resource is referenced in the definition. Resources may be referenced by name in the definition of a Task's steps.
<code>optional</code>	<code>boolean</code>	Optional declares the resource as optional. By default optional is set to false which makes a resource required. optional: true - the resource is considered optional optional: false - the resource is considered required (equivalent of not specifying it)
<code>targetPath</code>	<code>string</code>	TargetPath is the path in workspace directory where the resource will be copied.

Property	Type	Description
<code>type</code>	<code>string</code>	Type is the type of this resource;

`.spec.results`

Description

Results are values that this Task can output

Type

`array`

`.spec.results[]`

Description

TaskResult used to describe the results of a task

Type

`object`

Required

`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a human-readable description of the result
<code>name</code>	<code>string</code>	Name the given name

Property	Type	Description
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs results.
<code>type</code>	<code>string</code>	Type is the user-specified type of the result. The possible type is currently "string" and will support "array" in following work.
<code>value</code>	<code>object</code>	Value the expression used to retrieve the value of the result from an underlying Step.

`.spec.results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type

`object`

`.spec.results[].value`

Description

Value the expression used to retrieve the value of the result from an underlying Step.

Type

`object`

`.spec.sidecars`

Description

Sidecars are run alongside the Task's step containers. They begin before the steps start and end after the steps complete.

Type

array

.spec.sidecars[]

Description

Sidecar has nearly the same data structure as Step but does not have the ability to timeout.

Type

object

Required

name

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" vs. string literal "\$\$(VAR_NAME)". Escaped references will not be expanded, regardless of whether the variable exists. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the

Property	Type	Description
		will be unchanged. Double \$\$ are reduced to a sin allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(" Escaped references will never be expanded, regar whether the variable exists or not. Cannot be upda https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
env	array	List of environment variables to set in the Sidecar. updated.
envFrom	array	List of sources to populate environment variables i The keys defined within a source must be a C_IDE invalid keys will be reported as an event when the starting. When a key exists in multiple sources, the associated with the last source will take precedenc defined by an Env with a duplicate key will take pri Cannot be updated.
image	string	Image name to be used by the Sidecar. More info: https://kubernetes.io/docs/concepts/containers/images
imagePullPolicy	string	Image pull policy. One of Always, Never, IfNotPres to Always if :latest tag is specified, or IfNotPresent Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images

Property	Type	Description
lifecycle	object	Actions that the management system should take Sidecar lifecycle events. Cannot be updated.
livenessProbe	object	Periodic probe of Sidecar liveness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes
name	string	Name of the Sidecar specified as a DNS_LABEL. in a Task must have a unique name (DNS_LABEL). Cannot be updated.
ports	array	List of ports to expose from the Sidecar. Exposing ports gives the system additional information about the connections a container uses, but is primarily informational. Specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default address inside a container will be accessible from the host. Cannot be updated.
readinessProbe	object	Periodic probe of Sidecar service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes

Property	Type	Description
<code>resources</code>	<code>object</code>	Compute Resources required by this Sidecar. Can be updated. More info: https://kubernetes.io/docs/concepts/configuration/resources-containers/ ↗
<code>restartPolicy</code>	<code>string</code>	RestartPolicy refers to kubernetes RestartPolicy. It is set for an initContainer and must have its policy set. It is currently left optional to help support Kubernetes prior to 1.29 when this feature was introduced.
<code>script</code>	<code>string</code>	Script is the contents of an executable file to execute. If Script is not empty, the Step cannot have an Command or Args.
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Sidecar will be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>startupProbe</code>	<code>object</code>	StartupProbe indicates that the Pod the Sidecar is successfully initialized. If specified, no other probe is allowed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This is useful to provide different probe parameters at the beginning of the lifecycle, when it might take a long time to load data into the cache, than during steady-state operation. This can be updated. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/workloads/pod-lifecycle#container-probes ↗
<code>stdin</code>	<code>boolean</code>	Whether this Sidecar should allocate a buffer for stdin at container runtime. If this is not set, reads from stdin will always result in EOF. Default is false.
<code>stdinOnce</code>	<code>boolean</code>	Whether the container runtime should close the stdin stream after it has been opened by a single attach. When set to true, the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on pod start, is empty until the first client attaches to stdin, remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the pod is restarted. If this flag is false, a container process that reads from stdin will never receive an EOF. Default is false.
<code>terminationMessagePath</code>	<code>string</code>	Optional: Path at which the file to which the Sidecar's termination message will be written is mounted into the Sidecar's filesystem. Message written is intended to be brief final status of the container. Message will be truncated by the length of the path if greater than 4096 bytes. The total message length across all containers will be limited to 12kb. Defaults to /dev/termination-log. Cannot be updated.
<code>terminationMessagePolicy</code>	<code>string</code>	Indicate how the termination message should be populated. KillContainer will use the contents of terminationMessagePath to populate the Sidecar status message on both success and failure. FallbackToLogsOnError will use the last chunk of container log output if the termination message file is empty and the container log has been flushed. Can only be set on RestartPolicy Always. Default is KillContainer.

Property	Type	Description
		exited with an error. The log output is limited to 20 lines, whichever is smaller. Defaults to File. Cannot be updated.
<code>tty</code>	<code>boolean</code>	Whether this Sidecar should allocate a TTY for itself. Requires 'stdin' to be true. Default is false.
<code>volumeDevices</code>	<code>array</code>	volumeDevices is the list of block devices to be used by the Sidecar.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Sidecar's filesystem. Cannot be updated.
<code>workingDir</code>	<code>string</code>	Sidecar's working directory. If not specified, the container runtime's default will be used, which might be configurable for the container image. Cannot be updated.
<code>workspaces</code>	<code>array</code>	This is an alpha field. You must set the "enable-alpha" flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Sidecar wants exclusive access to. Adding a workspace to the list means that any other Step or Sidecar that does not request this Workspace will not have access to it.

`.spec.sidecars[].args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.spec.sidecars[].args[]`

Type

string

`.spec.sidecars[].command`

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Sidecar's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.spec.sidecars[].command[]`

Type

string

`.spec.sidecars[].env`

Description

List of environment variables to set in the Sidecar. Cannot be updated.

Type

array

`.spec.sidecars[].env[]`

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the environment variable. Must be a C_IDENTIFIER.
<code>value</code>	<code>string</code>	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".

Property	Type	Description
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.sidecars[].env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code> , <code>metadata.namespace</code> , <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , <code>spec.nodeName</code> , <code>spec.serviceAccountName</code> , <code>status.hostIP</code> , <code>status.podIP</code> , <code>status.podIPs</code> .
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (<code>limits.cpu</code> , <code>limits.memory</code> , <code>limits.ephemeral-storage</code> , <code>requests.cpu</code> , <code>requests.memory</code> and <code>requests.ephemeral-storage</code>) are currently supported.

Property	Type	Description
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.sidecars[].env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.sidecars[].env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`,

spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Type

object

Required

fieldPath

Property	Type	Description
apiVersion	string	Version of the schema the FieldPath is written in terms of, defaults to "v1".
fieldPath	string	Path of the field to select in the specified API version.

.spec.sidecars[].env[].valueFrom.resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
containerName	string	Container name: required for volumes, optional for env vars

Property	Type	Description
<code>divisor</code>	<code>integer</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.sidecars[].env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.sidecars[].envFrom`

Description

List of sources to populate environment variables in the Sidecar. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the Sidecar is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

array

.spec.sidecars[].envFrom[]

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

object

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

.spec.sidecars[].envFrom[].configMapRef

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.spec.sidecars[].envFrom[].secretRef`

Description

The Secret to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

`.spec.sidecars[].lifecycle`

Description

Actions that the management system should take in response to Sidecar lifecycle events.
Cannot be updated.

Type

object

Property	Type	Description
postStart	object	PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks
preStop	object	PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks

.spec.sidecars[].lifecycle.postStart

Description

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the

container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
exec	object	Exec specifies the action to take.
httpGet	object	HTTPGet specifies the http request to perform.
sleep	object	Sleep represents the duration that the container should sleep before being terminated.
tcpSocket	object	Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

.spec.sidecars[].lifecycle.postStart.exec

Description

Exec specifies the action to take.

Type

object

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].lifecycle.postStart.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.sidecars[].lifecycle.postStart.exec.command[]`

Type

`string`

`.spec.sidecars[].lifecycle.postStart.httpGet`

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
scheme	string	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].lifecycle.postStart.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

array

`.spec.sidecars[].lifecycle.postStart.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

object

Required

name

value

Property	Type	Description
name	string	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
value	string	The header field value

.spec.sidecars[].lifecycle.postStart.sleep

Description

Sleep represents the duration that the container should sleep before being terminated.

Type

object

Required

seconds

Property	Type	Description
seconds	integer	Seconds is the number of seconds to sleep.

.spec.sidecars[].lifecycle.postStart.tcpSocket

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.sidecars[].lifecycle.preStop

Description

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies the action to take.
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform.
<code>sleep</code>	<code>object</code>	Sleep represents the duration that the container should sleep before being terminated.
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

`.spec.sidecars[].lifecycle.preStop.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].lifecycle.preStop.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.sidecars[].lifecycle.preStop.exec.command[]`

Type

`string`

`.spec.sidecars[].lifecycle.preStop.httpGet`

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
scheme	string	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].lifecycle.preStop.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

array

`.spec.sidecars[].lifecycle.preStop.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

object

Required

name

value

Property	Type	Description
name	string	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
value	string	The header field value

.spec.sidecars[].lifecycle.preStop.sleep

Description

Sleep represents the duration that the container should sleep before being terminated.

Type

object

Required

seconds

Property	Type	Description
seconds	integer	Seconds is the number of seconds to sleep.

.spec.sidecars[].lifecycle.preStop.tcpSocket

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.sidecars[].livenessProbe

Description

Periodic probe of Sidecar liveness. Container will be restarted if the probe fails. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

object

Property	Type	Description
exec	object	Exec specifies the action to take.

Property	Type	Description
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies an action involving a GRPC
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCP Socket specifies an action involving a TCP
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown before it is killed. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are allowed to terminate gracefully. If specified, the "KILL" signal will be sent only if the grace period expires before the processes are allowed to terminate gracefully. Minimum value is 1 second.

Property	Type	Description
		pod are sent a termination signal and the timer processes are forcibly halted with a kill signal. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. A value of zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
timeoutSeconds	integer	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes

.spec.sidecars[].livenessProbe.exec

Description

Exec specifies the action to take.

Type

object

Property	Type	Description
command	array	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a

Property	Type	Description
		shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].livenessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

`.spec.sidecars[].livenessProbe.exec.command[]`

Type

string

`.spec.sidecars[].livenessProbe.grpc`

Description

GRPC specifies an action involving a GRPC port.

Type

object

Required

port

Property	Type	Description
<code>port</code>	<code>integer</code>	Port number of the gRPC service. Number must be in the range 1 to 65535.
<code>service</code>	<code>string</code>	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

`.spec.sidecars[].livenessProbe.httpGet`

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.

Property	Type	Description
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>	<code>int</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].livenessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.sidecars[].livenessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.sidecars[].livenessProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.sidecars[].ports`

Description

List of ports to expose from the Sidecar. Exposing a port here gives the system additional information about the network connections a container uses, but is primarily informational. Not specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default "0.0.0.0" address inside a container will be accessible from the network. Cannot be updated.

Type

array

.spec.sidecars[].ports[]

Description

ContainerPort represents a network port in a single container.

Type

object

Required

containerPort

Property	Type	Description
<code>containerPort</code>	<code>integer</code>	Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.
<code>hostIP</code>	<code>string</code>	What host IP to bind the external port to.
<code>hostPort</code>	<code>integer</code>	Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

Property	Type	Description
<code>name</code>	<code>string</code>	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.
<code>protocol</code>	<code>string</code>	Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

`.spec.sidecars[].readinessProbe`

Description

Periodic probe of Sidecar service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

`object`

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies the action to take.
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies an action involving a GRPC

Property	Type	Description
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a TCP port
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown before it is killed. The grace period is the duration in seconds after the processes running on the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It specifies that the pod won't attempt to create or restore data. Value must be non-negative integer. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal.

Property	Type	Description
		(no opportunity to shut down). This is a beta feature that requires enabling ProbeTerminationGracePeriodSeconds. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
timeoutSeconds	integer	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

.spec.sidecars[].readinessProbe.exec

Description

Exec specifies the action to take.

Type

object

Property	Type	Description
command	array	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

.spec.sidecars[].readinessProbe.exec.command

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.sidecars[].readinessProbe.exec.command[]

Type

string

.spec.sidecars[].readinessProbe.grpc

Description

GRPC specifies an action involving a GRPC port.

Type

object

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗).

Property	Type	Description
If this is not specified, the default behavior is defined by gRPC.		

`.spec.sidecars[].readinessProbe.httpGet`

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

Property	Type	Description
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.sidecars[].readinessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.sidecars[].readinessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.sidecars[].readinessProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.sidecars[].resources

Description

Compute Resources required by this Sidecar. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
claims	array	Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate.

Property	Type	Description
		This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.sidecars[].resources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

.spec.sidecars[].resources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
name	string	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.

.spec.sidecars[].resources.limits**Description**

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.sidecars[].resources.requests**Description**

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.sidecars[].securityContext**Description**

SecurityContext defines the security options the Sidecar should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

object

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: - run as Privileged - has <code>CAP_SYS_ADMIN</code> Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when <code>spec.os.name</code> is windows.

Property	Type	Description
<code>procMount</code>	<code>string</code>	<code>procMount</code> denotes the type of proc mount to use for the containers. The default is <code>DefaultProcMount</code> which uses the container runtime defaults for readonly paths and masked paths. This requires the <code>ProcMountType</code> feature flag to be enabled. Note that this field cannot be set when <code>spec.os.name</code> is <code>windows</code>.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when <code>spec.os.name</code> is <code>windows</code>.
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in <code>PodSecurityContext</code>. If set in both <code>SecurityContext</code> and <code>PodSecurityContext</code>, the value specified in <code>SecurityContext</code> takes precedence. Note that this field cannot be set when <code>spec.os.name</code> is <code>windows</code>.
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in <code>PodSecurityContext</code>. If set in

Property	Type	Description
		both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
runAsUser	integer	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
seLinuxOptions	object	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
seccompProfile	object	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.sidecars[].securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

`.spec.sidecars[].securityContext.capabilities.add`

Description

Added capabilities

Type

array

`.spec.sidecars[].securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.sidecars[].securityContext.capabilities.drop`

Description

Removed capabilities

Type

array

`.spec.sidecars[].securityContext.capabilities.drop[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.sidecars[].securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.spec.sidecars[].securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when `spec.os.name` is windows.

Type

object

Required

type

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile defined in a file on the node should be used. The profile must be

Property	Type	Description
		preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
<code>type</code>	<code>string</code>	type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

`.spec.sidecars[].securityContext.windowsOptions`

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

`object`

Property	Type	Description
<code>gmsaCredentialSpec</code>	<code>string</code>	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
<code>gmsaCredentialSpecName</code>	<code>string</code>	GMSACredentialSpecName is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

.spec.sidecars[].startupProbe

Description

StartupProbe indicates that the Pod the Sidecar is running in has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies the action to take.
<code>failureThreshold</code>	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	object	GRPC specifies an action involving a GRPC
<code>httpGet</code>	object	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	integer	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Property	Type	Description
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a TCP port. Probe must succeed in connecting to the port and then succeed in receiving at least one byte of data. This is useful for determining if a process is listening for incoming connections.
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully upon probe failure. The grace period is the duration in seconds after the processes running on the pod are sent a termination signal and the time given for the processes to forcibly halted with a kill signal. A value longer than the expected cleanup time may cause the pod to be terminated process. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used when unset.

Property	Type	Description
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.sidecars[].startupProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.sidecars[].startupProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.sidecars[].startupProbe.exec.command[]**Type**

string

.spec.sidecars[].startupProbe.grpc**Description**

GRPC specifies an action involving a GRPC port.

Type

object

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.sidecars[].startupProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
scheme	string	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.sidecars[].startupProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

array

.spec.sidecars[].startupProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

object

Required

name

value

Property	Type	Description
name	string	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
value	string	The header field value

.spec.sidecars[].startupProbe.tcpSocket

Description

TCPSocket specifies an action involving a TCP port.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>	<code>int</code>	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.sidecars[].volumeDevices`

Description

`volumeDevices` is the list of block devices to be used by the Sidecar.

Type

`array`

`.spec.sidecars[].volumeDevices[]`

Description

`volumeDevice` describes a mapping of a raw block device within a container.

Type

`object`

Required

`devicePath`

`name`

Property	Type	Description
<code>devicePath</code>	<code>string</code>	<code>devicePath</code> is the path inside of the container that the device will be mapped to.

Property	Type	Description
name	string	name must match the name of a persistentVolumeClaim in the pod

.spec.sidecars[].volumeMounts

Description

Volumes to mount into the Sidecar's filesystem. Cannot be updated.

Type

array

```
.spec.sidecars[].volumeMounts[]
```

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
mountPath	string	Path within the container at which the volume should be mounted. Must not contain '..'.

Property	Type	Description
<code>mountPropagation</code>	<code>string</code>	<code>mountPropagation</code> determines how mounts are propagated from the host to container and the other way around. When not set, <code>MountPropagationNone</code> is used. This field is beta in 1.10.
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to <code>SubPath</code> but environment variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. Defaults to "" (volume's root). <code>SubPathExpr</code> and <code>SubPath</code> are mutually exclusive.

.spec.sidecars[].workspaces

Description

This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Sidecar wants

exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

Type

array

.spec.sidecars[].workspaces[]

Description

WorkspaceUsage is used by a Step or Sidecar to declare that it wants isolated access to a Workspace defined in a Task.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath is the path that the workspace should be mounted to inside the Step or Sidecar, overriding any MountPath specified in the Task's WorkspaceDeclaration.
<code>name</code>	<code>string</code>	Name is the name of the workspace this Step or Sidecar wants access to.

.spec.stepTemplate

Description

StepTemplate can be used as the basis for all step containers within the Task, so that the steps inherit settings on the base container.

Type

object

Required

name

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The container's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$\$(VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
env	array	List of environment variables to set in the container. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell

Property	Type	Description
envFrom	array	List of sources to populate environment variables in the container. The keys defined within a source must be a C_IDENTIFIER. Invalid keys will be reported as an event when the container is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.
image	string	Default image name to use for each Step. More info: https://kubernetes.io/docs/concepts/containers/images . This field is optional to allow higher level config management to default or override container images in workload controllers Deployments and StatefulSets.
imagePullPolicy	string	Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images
lifecycle	object	Actions that the management system should take in response to container lifecycle events. Cannot be updated. Deprecated: This field will be removed in a future release.
livenessProbe	object	Periodic probe of container liveness. Container will be restarted if the probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes

Property	Type	Description
		Deprecated: This field will be removed in a future r
name	string	Default name for each Step specified as a DNS_L Step in a Task must have a unique name. Cannot Deprecated: This field will be removed in a future r
ports	array	List of ports to expose from the Step's container. E here gives the system additional information about connections a container uses, but is primarily infor specifying a port here DOES NOT prevent that poi exposed. Any port which is listening on the default address inside a container will be accessible from Cannot be updated. Deprecated: This field will be removed in a future r
readinessProbe	object	Periodic probe of container service readiness. Cor removed from service endpoints if the probe fails. updated. More info: https://kubernetes.io/docs/concepts/workloads/poc lifecycle#container-probes Deprecated: This field will be removed in a future r
resources	object	Compute Resources required by this Step. Cannot More info: https://kubernetes.io/docs/concepts/configuration/r resources-containers/

Property	Type	Description
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the Step will run with. If set, the fields of SecurityContext override equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>startupProbe</code>	<code>object</code>	DeprecatedStartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod is restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of the Pod's lifecycle, when it might take a long time to load data into a cache, than during steady-state operation. This can be updated. More info: https://kubernetes.io/docs/concepts/workloads/pods/lifecycle#container-probes ↗ Deprecated: This field will be removed in a future release.
<code>stdin</code>	<code>boolean</code>	Whether this Step should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin will always result in EOF. Default is false. Deprecated: This field will be removed in a future release.
<code>stdinOnce</code>	<code>boolean</code>	Whether the container runtime should close the stdin stream after it has been opened by a single attach. When set to true, the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on pod start, is empty until the first client attaches to stdin, and then remains open across subsequent attach sessions.

Property	Type	Description
		remains open and accepts data until the client disc which time stdin is closed and remains closed unti is restarted. If this flag is false, a container process from stdin will never receive an EOF. Default is fal Deprecated: This field will be removed in a future r
terminationMessagePath	string	Deprecated: This field will be removed in a future r cannot be meaningfully used.
terminationMessagePolicy	string	Deprecated: This field will be removed in a future r cannot be meaningfully used.
tty	boolean	Whether this Step should allocate a DeprecatedTT also requires 'stdin' to be true. Default is false. Deprecated: This field will be removed in a future r
volumeDevices	array	volumeDevices is the list of block devices to be us Step.
volumeMounts	array	Volumes to mount into the Step's filesystem. Cann
workingDir	string	Step's working directory. If not specified, the conta default will be used, which might be configured in t image. Cannot be updated.

.spec.stepTemplate.args

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.spec.stepTemplate.args[]

Type

string

.spec.stepTemplate.command

Description

Entrypoint array. Not executed within a shell. The OCI image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the Step's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.spec.stepTemplate.command[]

Type

string

.spec.stepTemplate.env

Description

List of environment variables to set in the container. Cannot be updated.

Type

array

.spec.stepTemplate.env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.
value	string	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never

Property	Type	Description
		be expanded, regardless of whether the variable exists or not. Defaults to "".
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.stepTemplate.env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Property	Type	Description
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.stepTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.stepTemplate.env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.stepTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (`limits.cpu`, `limits.memory`, `limits.ephemeral-storage`, `requests.cpu`, `requests.memory` and `requests.ephemeral-storage`) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.stepTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

.spec.stepTemplate.envFrom

Description

List of sources to populate environment variables in the Step. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the container is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

.spec.stepTemplate.envFrom[]

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from

Property	Type	Description
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.spec.stepTemplate.envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.spec.stepTemplate.envFrom[].secretRef`

Description

The Secret to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

.spec.stepTemplate.lifecycle

Description

Actions that the management system should take in response to container lifecycle events. Cannot be updated. Deprecated: This field will be removed in a future release.

Type

`object`

Property	Type	Description
<code>postStart</code>	<code>object</code>	PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗
<code>preStop</code>	<code>object</code>	PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before

Property	Type	Description
		the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗

.spec.stepTemplate.lifecycle.postStart

Description

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies the action to take.
<code>httpGet</code>	object	HTTPGet specifies the http request to perform.
<code>sleep</code>	object	Sleep represents the duration that the container should sleep before being terminated.

Property	Type	Description
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

`.spec.stepTemplate.lifecycle.postStart.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.stepTemplate.lifecycle.postStart.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.stepTemplate.lifecycle.postStart.exec.command[]

Type

string

.spec.stepTemplate.lifecycle.postStart.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an

Property	Type	Description
		IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.stepTemplate.lifecycle.postStart.httpGet.httpHeaderS`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.stepTemplate.lifecycle.postStart.httpGet.httpHeaderS[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

Property	Type	Description
<code>value</code>	<code>string</code>	The header field value

`.spec.stepTemplate.lifecycle.postStart.sleep`

Description

Sleep represents the duration that the container should sleep before being terminated.

Type

`object`

Required

`seconds`

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.spec.stepTemplate.lifecycle.postStart.tcpSocket`

Description

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>	<code>int</code>	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.stepTemplate.lifecycle.preStop`

Description

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

`object`

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies the action to take.
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform.

Property	Type	Description
<code>sleep</code>	<code>object</code>	Sleep represents the duration that the container should sleep before being terminated.
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

`.spec.stepTemplate.lifecycle.preStop.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.stepTemplate.lifecycle.preStop.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

`.spec.stepTemplate.lifecycle.preStop.exec.command[]`

Type

string

`.spec.stepTemplate.lifecycle.preStop.httpGet`

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.

Property	Type	Description
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>	<code>int</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.stepTemplate.lifecycle.preStop.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.stepTemplate.lifecycle.preStop.httpGet.httpHeaders[ ]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.stepTemplate.lifecycle.preStop.sleep`

Description

Sleep represents the duration that the container should sleep before being terminated.

Type

`object`

Required

`seconds`

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.spec.stepTemplate.lifecycle.preStop.tcpSocket`

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

Type

`object`

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.stepTemplate.livenessProbe

Description

Periodic probe of container liveness. Container will be restarted if the probe fails. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> Deprecated: This field will be removed in a future release.

Type

object

Property	Type	Description
exec	object	Exec specifies the action to take.
failureThreshold	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.

Property	Type	Description
<code>grpc</code>	<code>object</code>	GRPC specifies an action involving a GRPC
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a TCP
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown before it is killed. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It specifies that the pod needs to terminate gracefully within this time by default. If this value is nil, the pod's terminationGracePeriodSeconds will be used.

Property	Type	Description
		Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used when unset.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.stepTemplate.livenessProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.stepTemplate.livenessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.stepTemplate.livenessProbe.exec.command[]`

Type

`string`

`.spec.stepTemplate.livenessProbe.grpc`

Description

GRPC specifies an action involving a GRPC port.

Type

`object`

Required

`port`

Property	Type	Description
<code>port</code>	<code>integer</code>	Port number of the gRPC service. Number must be in the range 1 to 65535.

Property	Type	Description
<code>service</code>	<code>string</code>	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.stepTemplate.livenessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.

Property	Type	Description
<code>port</code>	<code>integer</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.stepTemplate.livenessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.stepTemplate.livenessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name` `value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

Property	Type	Description
<code>value</code>	<code>string</code>	The header field value

`.spec.stepTemplate.livenessProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.stepTemplate.ports`

Description

List of ports to expose from the Step's container. Exposing a port here gives the system additional information about the network connections a container uses, but is primarily informational. Not specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default "0.0.0.0" address inside a container will be

accessible from the network. Cannot be updated. Deprecated: This field will be removed in a future release.

Type

array

.spec.stepTemplate.ports[]

Description

ContainerPort represents a network port in a single container.

Type

object

Required

containerPort

Property	Type	Description
containerPort	integer	Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.
hostIP	string	What host IP to bind the external port to.
hostPort	integer	Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.

Property	Type	Description
<code>name</code>	<code>string</code>	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.
<code>protocol</code>	<code>string</code>	Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

.spec.stepTemplate.readinessProbe

Description

Periodic probe of container service readiness. Container will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Deprecated: This field will be removed in a future release.

Type

`object`

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies the action to take.
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.

Property	Type	Description
<code>grpc</code>	<code>object</code>	GRPC specifies an action involving a GRPC
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCP Socket specifies an action involving a TCP
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown before it is killed. During this period, the pod can still receive requests, and exit codes for the processes running within the pod will be returned, and the termination signal will be sent. If the grace period is nil, the pod's terminationGracePeriodSeconds will be used.

Property	Type	Description
		Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used when unset.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More information: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.stepTemplate.readinessProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.stepTemplate.readinessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.stepTemplate.readinessProbe.exec.command[]`

Type

`string`

`.spec.stepTemplate.readinessProbe.grpc`

Description

GRPC specifies an action involving a GRPC port.

Type

`object`

Required

`port`

Property	Type	Description
<code>port</code>	<code>integer</code>	Port number of the gRPC service. Number must be in the range 1 to 65535.

Property	Type	Description
<code>service</code>	<code>string</code>	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.stepTemplate.readinessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.

Property	Type	Description
<code>port</code>	<code>integer</code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.stepTemplate.readinessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.stepTemplate.readinessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.

Property	Type	Description
<code>value</code>	<code>string</code>	The header field value

`.spec.stepTemplate.readinessProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.stepTemplate.resources`

Description

Compute Resources required by this Step. Cannot be updated. More info:
<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

`object`

Property	Type	Description
		Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.
claims	array	This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.stepTemplate.resources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

.spec.stepTemplate.resources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
name	string	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.

.spec.stepTemplate.resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.stepTemplate.resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.stepTemplate.securityContext

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info:

<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

object

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: - run as Privileged - has <code>CAP_SYS_ADMIN</code> Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when <code>spec.os.name</code> is windows.

Property	Type	Description
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>procMount</code>	<code>string</code>	<code>procMount</code> denotes the type of proc mount to use for the containers. The default is <code>DefaultProcMount</code> which uses the container runtime defaults for readonly paths and masked paths. This requires the <code>ProcMountType</code> feature flag to be enabled. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>readOnlyRootFilesystem</code>	<code>boolean</code>	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when <code>spec.os.name</code> is windows.
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in <code>PodSecurityContext</code> . If set in both <code>SecurityContext</code> and <code>PodSecurityContext</code> , the value specified in <code>SecurityContext</code> takes precedence. Note that this field cannot be set when <code>spec.os.name</code> is windows.

Property	Type	Description
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.stepTemplate.securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities

Property	Type	Description
<code>drop</code>	<code>array</code>	Removed capabilities

`.spec.stepTemplate.securityContext.capabilities.add`

Description

Added capabilities

Type

`array`

`.spec.stepTemplate.securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

`string`

`.spec.stepTemplate.securityContext.capabilities.drop`

Description

Removed capabilities

Type

`array`

`.spec.stepTemplate.securityContext.capabilities.drop[]`

Description

Capability represent POSIX capabilities type

Type

`string`

`.spec.stepTemplate.securityContext.seLinuxOptions`

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.spec.stepTemplate.securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.

Type

object

Required

type

Property	Type	Description
localhostProfile	string	localhostProfile indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
type	string	type indicates which kind of seccomp profile will be applied. Valid options are: Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.spec.stepTemplate.securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
<code>gmsaCredentialSpec</code>	<code>string</code>	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
<code>gmsaCredentialSpecName</code>	<code>string</code>	GMSACredentialSpecName is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

.spec.stepTemplate.startupProbe

Description

DeprecatedStartupProbe indicates that the Pod has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Deprecated: This field will be removed in a future release.

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies the action to take.
<code>failureThreshold</code>	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	object	GRPC specifies an action involving a GRPC
<code>httpGet</code>	object	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	integer	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Property	Type	Description
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a TCP port. Probe must succeed in connecting to the port and then succeed in receiving data. The default is <code>TCPSocket</code> .
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully upon probe failure. The grace period is the duration in seconds after the processes running on the pod are sent a termination signal and the time window within which the pod must terminate. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used.

Property	Type	Description
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes

`.spec.stepTemplate.startupProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.stepTemplate.startupProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.stepTemplate.startupProbe.exec.command[]**Type**

string

.spec.stepTemplate.startupProbe.grpc**Description**

GRPC specifies an action involving a GRPC port.

Type

object

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.stepTemplate.startupProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
scheme	string	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.stepTemplate.startupProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

array

.spec.stepTemplate.startupProbe.httpGet.httpHeaders[]

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

object

Required

name

value

Property	Type	Description
name	string	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
value	string	The header field value

.spec.stepTemplate.startupProbe.tcpSocket

Description

TCPSocket specifies an action involving a TCP port.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>	<code>int</code>	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.stepTemplate.volumeDevices`

Description

`volumeDevices` is the list of block devices to be used by the Step.

Type

`array`

`.spec.stepTemplate.volumeDevices[]`

Description

`volumeDevice` describes a mapping of a raw block device within a container.

Type

`object`

Required

`devicePath`

`name`

Property	Type	Description
<code>devicePath</code>	<code>string</code>	<code>devicePath</code> is the path inside of the container that the device will be mapped to.

Property	Type	Description
name	string	name must match the name of a persistentVolumeClaim in the pod

.spec.stepTemplate.volumeMounts

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type

array

```
.spec.stepTemplate.volumeMounts[]
```

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'.

Property	Type	Description
<code>mountPropagation</code>	<code>string</code>	<code>mountPropagation</code> determines how mounts are propagated from the host to container and the other way around. When not set, <code>MountPropagationNone</code> is used. This field is beta in 1.10.
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to <code>SubPath</code> but environment variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. Defaults to "" (volume's root). <code>SubPathExpr</code> and <code>SubPath</code> are mutually exclusive.

.spec.steps

Description

Steps are the steps of the build; each step is run sequentially with the source mounted into `/workspace`.

Type

array

.spec.steps[]

Description

Step runs a subcomponent of a Task

Type

object

Required

name

Property	Type	Description
args	array	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code> , which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code> . Escaped references will never be expanded, regardless of whether the variable exists. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code> , which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code> . Escaped references will never be expanded, regardless of whether the variable exists. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell

Property	Type	Description
		whether the variable exists or not. Cannot be updated. https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell
<code>env</code>	<code>array</code>	List of environment variables to set in the container. Cannot be updated.
<code>envFrom</code>	<code>array</code>	List of sources to populate environment variables in the container. The keys defined within a source must be unique. The keys must be in the format C_IDENTIFIER. All invalid keys will be reported as an error when the container is starting. When a key exists in multiple sources, the value associated with the last source takes precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.
<code>image</code>	<code>string</code>	Image reference name to run for this Step. More info: https://kubernetes.io/docs/concepts/containers/images
<code>imagePullPolicy</code>	<code>string</code>	Image pull policy. One of Always, Never, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info: https://kubernetes.io/docs/concepts/containers/images

Property	Type	Description
<code>lifecycle</code>	<code>object</code>	Actions that the management system should take container lifecycle events. Cannot be updated. Deprecated: This field will be removed in a future r
<code>livenessProbe</code>	<code>object</code>	Periodic probe of container liveness. Step will be r probe fails. Cannot be updated. More info: https://kubernetes.io/docs/concepts/workloads/poc lifecycle#container-probes ↗ Deprecated: This field will be removed in a future r
<code>name</code>	<code>string</code>	Name of the Step specified as a DNS_LABEL. Ea Task must have a unique name.
<code>onError</code>	<code>string</code>	OnError defines the exiting behavior of a container be set to [continue stopAndFail]
<code>params</code>	<code>array</code>	Params declares parameters passed to this step a
<code>ports</code>	<code>array</code>	List of ports to expose from the Step's container. E here gives the system additional information about connections a container uses, but is primarily infor specifying a port here DOES NOT prevent that poi exposed. Any port which is listening on the default address inside a container will be accessible from Cannot be updated.

Property	Type	Description
		Deprecated: This field will be removed in a future r
readinessProbe	object	Periodic probe of container service readiness. Step removed from service endpoints if the probe fails. updated. More info: https://kubernetes.io/docs/concepts/workloads/poc lifecycle#container-probes ↗ Deprecated: This field will be removed in a future r
ref	object	Contains the reference to an existing StepAction.
resources	object	Compute Resources required by this Step. Cannot More info: https://kubernetes.io/docs/concepts/configuration/r resources-containers/ ↗
results	array	Results declares StepResults produced by the Ste This is field is at an ALPHA stability level and gate step-actions" feature flag. It can be used in an inlined Step when used to stor \$(step.results.resultName.path). It cannot be used referencing StepActions using [v1beta1.Step.Ref]. declared by the StepActions will be stored here ins
script	string	Script is the contents of an executable file to exec

Property	Type	Description
		If Script is not empty, the Step cannot have an Cor the Args will be passed to the Script.
<code>securityContext</code>	<code>object</code>	SecurityContext defines the security options the S run with. If set, the fields of SecurityContext overri equivalent fields of PodSecurityContext. More info https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗
<code>startupProbe</code>	<code>object</code>	DeprecatedStartupProbe indicates that the Pod thi has successfully initialized. If specified, no other p executed until this completes successfully. If this p Pod will be restarted, just as if the livenessProbe f be used to provide different probe parameters at tl a Pod's lifecycle, when it might take a long time to warm a cache, than during steady-state operation. be updated. More info: https://kubernetes.io/docs/concepts/workloads/poc lifecycle#container-probes ↗ Deprecated: This field will be removed in a future r
<code>stderrConfig</code>	<code>object</code>	Stores configuration for the stderr stream of the str
<code>stdin</code>	<code>boolean</code>	Whether this container should allocate a buffer for container runtime. If this is not set, reads from stdi container will always result in EOF. Default is false Deprecated: This field will be removed in a future r

Property	Type	Description
<code>stdinOnce</code>	<code>boolean</code>	Whether the container runtime should close the stdin stream after it has been opened by a single attach. When the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on pod start, is empty until the first client attaches to stdin, remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the pod is restarted. If this flag is false, a container process must start from stdin will never receive an EOF. Default is false Deprecated: This field will be removed in a future release
<code>stdoutConfig</code>	<code>object</code>	Stores configuration for the stdout stream of the step
<code>terminationMessagePath</code>	<code>string</code>	Deprecated: This field will be removed in a future release and can't be meaningfully used.
<code>terminationMessagePolicy</code>	<code>string</code>	Deprecated: This field will be removed in a future release and can't be meaningfully used.
<code>timeout</code>	<code>string</code>	Timeout is the time after which the step times out. The value must never be empty. Refer to Go's ParseDuration documentation for the format: https://golang.org/pkg/time/#ParseDuration

Property	Type	Description
<code>tty</code>	<code>boolean</code>	Whether this container should allocate a Deprecat itself, also requires 'stdin' to be true. Default is fals Deprecated: This field will be removed in a future r
<code>volumeDevices</code>	<code>array</code>	volumeDevices is the list of block devices to be us Step.
<code>volumeMounts</code>	<code>array</code>	Volumes to mount into the Step's filesystem. Cann
<code>when</code>	<code>array</code>	WhenExpressions are used to specify whether a T executed or skipped All of them need to evaluate t guarded Task to be executed.
<code>workingDir</code>	<code>string</code>	Step's working directory. If not specified, the conta default will be used, which might be configured in t image. Cannot be updated.
<code>workspaces</code>	<code>array</code>	This is an alpha field. You must set the "enable-ap flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task t wants exclusive access to. Adding a workspace to that any other Step or Sidecar that does not also r Workspace will not have access to it.

`.spec.steps[].args`

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.spec.steps[].args[]`

Type

string

`.spec.steps[].command`

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `"$$$(VAR_NAME)"` will produce the string literal `"$(VAR_NAME)"`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

`.spec.steps[].command[]`

Type

string

.spec.steps[].env

Description

List of environment variables to set in the container. Cannot be updated.

Type

array

.spec.steps[].env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.
value	string	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\$(VAR_NAME)". Escaped references will never

Property	Type	Description
		be expanded, regardless of whether the variable exists or not. Defaults to "".
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.steps[].env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Property	Type	Description
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.steps[].env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.steps[].env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.steps[].env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (`limits.cpu`, `limits.memory`, `limits.ephemeral-storage`, `requests.cpu`, `requests.memory` and `requests.ephemeral-storage`) are currently supported.

Type

`object`**Required**`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.steps[].env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type`object`**Required**`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.steps[].envFrom`

Description

List of sources to populate environment variables in the container. The keys defined within a source must be a C_IDENTIFIER. All invalid keys will be reported as an event when the container is starting. When a key exists in multiple sources, the value associated with the last source will take precedence. Values defined by an Env with a duplicate key will take precedence. Cannot be updated.

Type

`array`

`.spec.steps[].envFrom[]`

Description

EnvFromSource represents the source of a set of ConfigMaps

Type

`object`

Property	Type	Description
<code>configMapRef</code>	<code>object</code>	The ConfigMap to select from

Property	Type	Description
<code>prefix</code>	<code>string</code>	An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.
<code>secretRef</code>	<code>object</code>	The Secret to select from

`.spec.steps[].envFrom[].configMapRef`

Description

The ConfigMap to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap must be defined

`.spec.steps[].envFrom[].secretRef`

Description

The Secret to select from

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret must be defined

.spec.steps[].lifecycle

Description

Actions that the management system should take in response to container lifecycle events. Cannot be updated. Deprecated: This field will be removed in a future release.

Type

`object`

Property	Type	Description
<code>postStart</code>	<code>object</code>	PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks ↗
<code>preStop</code>	<code>object</code>	PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before

Property	Type	Description
		the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info: https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks

.spec.steps[].lifecycle.postStart

Description

PostStart is called immediately after a container is created. If the handler fails, the container is terminated and restarted according to its restart policy. Other management of the container blocks until the hook completes. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
exec	object	Exec specifies the action to take.
httpGet	object	HTTPGet specifies the http request to perform.
sleep	object	Sleep represents the duration that the container should sleep before being terminated.

Property	Type	Description
<code>tcpSocket</code>	<code>object</code>	Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

`.spec.steps[].lifecycle.postStart.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.steps[].lifecycle.postStart.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.steps[].lifecycle.postStart.exec.command[]

Type

string

.spec.steps[].lifecycle.postStart.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an

Property	Type	Description
		IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.steps[].lifecycle.postStart.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.steps[].lifecycle.postStart.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.steps[].lifecycle.postStart.sleep`

Description

Sleep represents the duration that the container should sleep before being terminated.

Type

object

Required

seconds

Property	Type	Description
<code>seconds</code>	<code>integer</code>	Seconds is the number of seconds to sleep.

`.spec.steps[].lifecycle.postStart.tcpSocket`

Description

Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an

Property	Type	Description
		IANA_SVC_NAME.

`.spec.steps[].lifecycle.preStop`

Description

PreStop is called immediately before a container is terminated due to an API request or management event such as liveness/startup probe failure, preemption, resource contention, etc. The handler is not called if the container crashes or exits. The Pod's termination grace period countdown begins before the PreStop hook is executed. Regardless of the outcome of the handler, the container will eventually terminate within the Pod's termination grace period (unless delayed by finalizers). Other management of the container blocks until the hook completes or until the termination grace period is reached. More info:

<https://kubernetes.io/docs/concepts/containers/container-lifecycle-hooks/#container-hooks>

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies the action to take.
<code>httpGet</code>	object	HTTPGet specifies the http request to perform.
<code>sleep</code>	object	Sleep represents the duration that the container should sleep before being terminated.
<code>tcpSocket</code>	object	Deprecated. TCPSocket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There

Property	Type	Description
		are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

`.spec.steps[].lifecycle.preStop.exec`

Description

Exec specifies the action to take.

Type

object

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.steps[].lifecycle.preStop.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

`.spec.steps[].lifecycle.preStop.exec.command[]`

Type

`string`

`.spec.steps[].lifecycle.preStop.httpGet`

Description

HTTPGet specifies the http request to perform.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

Property	Type	Description
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.steps[].lifecycle.preStop.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.steps[].lifecycle.preStop.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.steps[].lifecycle.preStop.sleep`

Description

Sleep represents the duration that the container should sleep before being terminated.

Type

object

Required

seconds

Property	Type	Description
seconds	integer	Seconds is the number of seconds to sleep.

.spec.steps[].lifecycle.preStop.tcpSocket

Description

Deprecated. TCP Socket is NOT supported as a LifecycleHandler and kept for the backward compatibility. There are no validation of this field and lifecycle hooks will fail in runtime when tcp handler is specified.

Type

object

Required

port

Property	Type	Description
host	string	Optional: Host name to connect to, defaults to the pod IP.
port		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

.spec.steps[].livenessProbe

Description

Periodic probe of container liveness. Step will be restarted if the probe fails. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> **Deprecated:** This field will be removed in a future release.

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies the action to take.
<code>failureThreshold</code>	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	object	GRPC specifies an action involving a GRPC
<code>httpGet</code>	object	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	integer	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes

Property	Type	Description
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a TCP port. Probe must succeed in connecting to the port and then immediately receive data. The port number is expected to be specified in the <code>port</code> field of the <code>TCPSocket</code> object.
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs to gracefully shutdown upon probe failure. The grace period is the duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It specifies that the pod needs to terminate gracefully. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unspecified.

Property	Type	Description
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/liveness-probe/#configuration

`.spec.steps[].livenessProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.steps[].livenessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

array

.spec.steps[].livenessProbe.exec.command[]**Type**

string

.spec.steps[].livenessProbe.grpc**Description**

GRPC specifies an action involving a GRPC port.

Type

object

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.steps[].livenessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.
httpHeaders	array	Custom headers to set in the request. HTTP allows repeated headers.
path	string	Path to access on the HTTP server.
port		Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
scheme	string	Scheme to use for connecting to the host. Defaults to HTTP.

.spec.steps[].livenessProbe.httpGet.httpHeaders

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

array

`.spec.steps[].livenessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

object

Required

name

value

Property	Type	Description
name	string	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
value	string	The header field value

`.spec.steps[].livenessProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

object

Required

port

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>	<code>int</code>	Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.steps[].params`

Description

Params declares parameters passed to this step action.

Type

`array`

`.spec.steps[].params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>	<code>object</code>	ParamValue is a type that can hold a single string or string array. Used in JSON unmarshalling so that a single JSON field can

Property	Type	Description
		accept either an individual string or an array of strings.

`.spec.steps[].params[].value`

Description

ParamValue is a type that can hold a single string or string array. Used in JSON unmarshalling so that a single JSON field can accept either an individual string or an array of strings.

Type

object

`.spec.steps[].ports`

Description

List of ports to expose from the Step's container. Exposing a port here gives the system additional information about the network connections a container uses, but is primarily informational. Not specifying a port here DOES NOT prevent that port from being exposed. Any port which is listening on the default "0.0.0.0" address inside a container will be accessible from the network. Cannot be updated. Deprecated: This field will be removed in a future release.

Type

array

`.spec.steps[].ports[]`

Description

ContainerPort represents a network port in a single container.

Type

object

Required

`containerPort`

Property	Type	Description
<code>containerPort</code>	<code>integer</code>	Number of port to expose on the pod's IP address. This must be a valid port number, $0 < x < 65536$.
<code>hostIP</code>	<code>string</code>	What host IP to bind the external port to.
<code>hostPort</code>	<code>integer</code>	Number of port to expose on the host. If specified, this must be a valid port number, $0 < x < 65536$. If HostNetwork is specified, this must match ContainerPort. Most containers do not need this.
<code>name</code>	<code>string</code>	If specified, this must be an IANA_SVC_NAME and unique within the pod. Each named port in a pod must have a unique name. Name for the port that can be referred to by services.
<code>protocol</code>	<code>string</code>	Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

`.spec.steps[].readinessProbe`

Description

Periodic probe of container service readiness. Step will be removed from service endpoints if the probe fails. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes>

Deprecated: This field will be removed in a future release.

Type

object

Property	Type	Description
<code>exec</code>	object	Exec specifies the action to take.
<code>failureThreshold</code>	integer	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	object	GRPC specifies an action involving a GRPC
<code>httpGet</code>	object	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	integer	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/controllers/liveness-probe/#container-probes
<code>periodSeconds</code>	integer	How often (in seconds) to perform the probe. Default value is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	integer	Minimum consecutive successes for the probe to be considered successful after having failed. Default value is 1. Must be 1 for liveness and startup. Minimum value is 1.

Property	Type	Description
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a T
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs gracefully upon probe failure. The grace period duration in seconds after the processes running in the pod are sent a termination signal and the time when the processes are forcibly halted with a kill signal. It specifies that the pod should not be restarted if it is terminated. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided by the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/lifecycle#container-probes

`.spec.steps[].readinessProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.steps[].readinessProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.steps[].readinessProbe.exec.command[]`

Type

`string`

`.spec.steps[].readinessProbe.grpc`

Description

GRPC specifies an action involving a GRPC port.

Type

`object`

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.steps[].readinessProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

Property	Type	Description
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>	<code></code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.steps[].readinessProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.steps[].readinessProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.steps[].readinessProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.steps[].ref`

Description

Contains the reference to an existing StepAction.

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referenced step
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.steps[].ref.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

array

`.spec.steps[].ref.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value	object	ParamValue is a type that can hold a single string or string array. Used in JSON unmarshalling so that a single JSON field can accept either an individual string or an array of strings.

.spec.steps[].ref.params[].value

Description

ParamValue is a type that can hold a single string or string array. Used in JSON unmarshalling so that a single JSON field can accept either an individual string or an array of strings.

Type

object

.spec.steps[].resources

Description

Compute Resources required by this Step. Cannot be updated. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

Property	Type	Description
claims	array	Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container.

Property	Type	Description
		This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.
limits	object	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
requests	object	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

.spec.steps[].resources.claims

Description

Claims lists the names of resources, defined in spec.resourceClaims, that are used by this container. This is an alpha field and requires enabling the DynamicResourceAllocation feature gate. This field is immutable. It can only be set for containers.

Type

array

.spec.steps[].resources.claims[]

Description

ResourceClaim references one entry in PodSpec.ResourceClaims.

Type

object

Required

name

Property	Type	Description
name	string	Name must match the name of one entry in pod.spec.resourceClaims of the Pod where this field is used. It makes that resource available inside a container.

.spec.steps[].resources.limits

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.steps[].resources.requests

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

.spec.steps[].results

Description

Results declares StepResults produced by the Step. This field is at an ALPHA stability level and gated by "enable-step-actions" feature flag. It can be used in an inlined Step when used to store Results to \$(step.results.resultName.path). It cannot be used when referencing StepActions using [v1beta1.Step.Ref]. The Results declared by the StepActions will be stored here instead.

Type

array

.spec.steps[].results[]

Description

StepResult used to describe the Results of a Step. This field is at an BETA stability level and gated by "enable-step-actions" feature flag.

Type

object

Required

name

Property	Type	Description
description	string	Description is a human-readable description of the result
name	string	Name the given name
properties	object	Properties is the JSON Schema properties to support key-value pairs results.

Property	Type	Description
<code>type</code>	<code>string</code>	The possible types are 'string', 'array', and 'object', with 'string' as the default.

`.spec.steps[].results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type

`object`

`.spec.steps[].securityContext`

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: <https://kubernetes.io/docs/tasks/configure-pod-container/security-context/>

Type

`object`

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: run as Privileged

Property	Type	Description
		2. has CAP_SYS_ADMIN Note that this field cannot be set when spec.os.name is windows.
capabilities	object	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.
privileged	boolean	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to false. Note that this field cannot be set when spec.os.name is windows.
procMount	string	procMount denotes the type of proc mount to use for the containers. The default is DefaultProcMount which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
readOnlyRootFilesystem	boolean	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>runAsGroup</code>	<code>integer</code>	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>runAsNonRoot</code>	<code>boolean</code>	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
<code>runAsUser</code>	<code>integer</code>	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Property	Type	Description
<code>seLinuxOptions</code>	<code>object</code>	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
<code>seccompProfile</code>	<code>object</code>	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.steps[].securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
add	array	Added capabilities
drop	array	Removed capabilities

`.spec.steps[].securityContext.capabilities.add`

Description

Added capabilities

Type

array

`.spec.steps[].securityContext.capabilities.add[]`

Description

Capability represent POSIX capabilities type

Type

string

`.spec.steps[].securityContext.capabilities.drop`

Description

Removed capabilities

Type

array

.spec.steps[].securityContext.capabilities.drop[]

Description

Capability represent POSIX capabilities type

Type

string

.spec.steps[].securityContext.seLinuxOptions

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
level	string	Level is SELinux level label that applies to the container.
role	string	Role is a SELinux role label that applies to the container.
type	string	Type is a SELinux type label that applies to the container.

Property	Type	Description
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.spec.steps[].securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when `spec.os.name` is `windows`.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	<code>localhostProfile</code> indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.

Property	Type	Description
		type indicates which kind of seccomp profile will be applied. Valid options are:
type	string	Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.spec.steps[].securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
gmsaCredentialSpec	string	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.
gmsaCredentialSpecName	string	GMSACredentialSpecName is the name of the GMSA credential spec to use.

Property	Type	Description
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

`.spec.steps[].startupProbe`

Description

DeprecatedStartupProbe indicates that the Pod this Step runs in has successfully initialized. If specified, no other probes are executed until this completes successfully. If this probe fails, the Pod will be restarted, just as if the livenessProbe failed. This can be used to provide different probe parameters at the beginning of a Pod's lifecycle, when it might take a long time to load data or warm a cache, than during steady-state operation. This cannot be updated. More info: <https://kubernetes.io/docs/concepts/workloads/pods/pod-lifecycle#container-probes> Deprecated: This field will be removed in a future release.

Type

`object`

Property	Type	Description
<code>exec</code>	<code>object</code>	Exec specifies the action to take.
<code>failureThreshold</code>	<code>integer</code>	Minimum consecutive failures for the probe to be considered failed after having succeeded. Default value is 3. Minimum value is 1.
<code>grpc</code>	<code>object</code>	GRPC specifies an action involving a GRPC
<code>httpGet</code>	<code>object</code>	HTTPGet specifies the http request to perform
<code>initialDelaySeconds</code>	<code>integer</code>	Number of seconds after the container has started before liveness probes are initiated. More info: https://kubernetes.io/docs/concepts/workloads/controllers/lifecycle#container-probes
<code>periodSeconds</code>	<code>integer</code>	How often (in seconds) to perform the probe. Default is 10 seconds. Minimum value is 1.
<code>successThreshold</code>	<code>integer</code>	Minimum consecutive successes for the probe to be considered successful after having failed. Default is 1. Must be 1 for liveness and startup. Minimum value is 1.
<code>tcpSocket</code>	<code>object</code>	TCPSocket specifies an action involving a TCP

Property	Type	Description
<code>terminationGracePeriodSeconds</code>	<code>integer</code>	Optional duration in seconds the pod needs gracefully upon probe failure. The grace period duration in seconds after the processes running in the pod are sent a termination signal and the timer begins. Once the grace period has elapsed, the processes are forcibly halted with a kill signal. A value greater than zero indicates a grace period; a value of zero indicates no grace period. If this value is nil, the pod's terminationGracePeriodSeconds will be used. Otherwise, this value overrides the value provided in the pod spec. Value must be non-negative integer. Value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta feature and requires enabling ProbeTerminationGracePeriod. Minimum value is 1. spec.terminationGracePeriodSeconds is used.
<code>timeoutSeconds</code>	<code>integer</code>	Number of seconds after which the probe times out. Defaults to 1 second. Minimum value is 1. More info: https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/#liveness-probe

`.spec.steps[].startupProbe.exec`

Description

Exec specifies the action to take.

Type

`object`

Property	Type	Description
<code>command</code>	<code>array</code>	Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions (' ', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

`.spec.steps[].startupProbe.exec.command`

Description

Command is the command line to execute inside the container, the working directory for the command is root ('/') in the container's filesystem. The command is simply exec'd, it is not run inside a shell, so traditional shell instructions ('|', etc) won't work. To use a shell, you need to explicitly call out to that shell. Exit status of 0 is treated as live/healthy and non-zero is unhealthy.

Type

`array`

`.spec.steps[].startupProbe.exec.command[]`

Type

`string`

`.spec.steps[].startupProbe.grpc`

Description

GRPC specifies an action involving a GRPC port.

Type

`object`

Required

port

Property	Type	Description
port	integer	Port number of the gRPC service. Number must be in the range 1 to 65535.
service	string	Service is the name of the service to place in the gRPC HealthCheckRequest (see https://github.com/grpc/grpc/blob/master/doc/health-checking.md ↗). If this is not specified, the default behavior is defined by gRPC.

.spec.steps[].startupProbe.httpGet

Description

HTTPGet specifies the http request to perform.

Type

object

Required

port

Property	Type	Description
host	string	Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

Property	Type	Description
<code>httpHeaders</code>	<code>array</code>	Custom headers to set in the request. HTTP allows repeated headers.
<code>path</code>	<code>string</code>	Path to access on the HTTP server.
<code>port</code>	<code></code>	Name or number of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.
<code>scheme</code>	<code>string</code>	Scheme to use for connecting to the host. Defaults to HTTP.

`.spec.steps[].startupProbe.httpGet.httpHeaders`

Description

Custom headers to set in the request. HTTP allows repeated headers.

Type

`array`

`.spec.steps[].startupProbe.httpGet.httpHeaders[]`

Description

HTTPHeader describes a custom header to be used in HTTP probes

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	The header field name. This will be canonicalized upon output, so case-variant names will be understood as the same header.
<code>value</code>	<code>string</code>	The header field value

`.spec.steps[].startupProbe.tcpSocket`

Description

TCPSocket specifies an action involving a TCP port.

Type

`object`

Required

`port`

Property	Type	Description
<code>host</code>	<code>string</code>	Optional: Host name to connect to, defaults to the pod IP.
<code>port</code>		Number or name of the port to access on the container. Number must be in the range 1 to 65535. Name must be an IANA_SVC_NAME.

`.spec.steps[].stderrConfig`

Description

Stores configuration for the stderr stream of the step.

Type

object

Property	Type	Description
path	string	Path to duplicate stdout stream to on container's local filesystem.

.spec.steps[].stdoutConfig

Description

Stores configuration for the stdout stream of the step.

Type

object

Property	Type	Description
path	string	Path to duplicate stdout stream to on container's local filesystem.

.spec.steps[].volumeDevices

Description

volumeDevices is the list of block devices to be used by the Step.

Type

array

.spec.steps[].volumeDevices[]

Description

volumeDevice describes a mapping of a raw block device within a container.

Type

`object`**Required**`devicePath``name`

Property	Type	Description
<code>devicePath</code>	<code>string</code>	devicePath is the path inside of the container that the device will be mapped to.
<code>name</code>	<code>string</code>	name must match the name of a persistentVolumeClaim in the pod

`.spec.steps[].volumeMounts`

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type`array`

`.spec.steps[].volumeMounts[]`

Description

VolumeMount describes a mounting of a Volume within a container.

Type`object`**Required**`mountPath``name`

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'.
<code>mountPropagation</code>	<code>string</code>	<code>mountPropagation</code> determines how mounts are propagated from the host to container and the other way around. When not set, <code>MountPropagationNone</code> is used. This field is beta in 1.10.
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to <code>SubPath</code> but environment variable references <code>\$(VAR_NAME)</code> are expanded using the container's environment. Defaults to "" (volume's root). <code>SubPathExpr</code> and <code>SubPath</code> are mutually exclusive.

`.spec.steps[].when`

Description

WhenExpressions are used to specify whether a Task should be executed or skipped All of them need to evaluate to True for a guarded Task to be executed.

Type

array

.spec.steps[].when[]

Description

WhenExpression allows a PipelineTask to declare expressions to be evaluated before the Task is run to determine whether the Task should be executed or skipped

Type

object

Property	Type	Description
<code>cel</code>	<code>string</code>	CEL is a string of Common Language Expression, which can be used to conditionally execute the task based on the result of the expression evaluation More info about CEL syntax: https://github.com/google/cel-spec/blob/master/doc/langdef.md ↗
<code>input</code>	<code>string</code>	Input is the string for guard checking which can be a static input or an output from a parent Task
<code>operator</code>	<code>string</code>	Operator that represents an Input's relationship to the values
<code>values</code>	<code>array</code>	Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

`.spec.steps[].when[].values`

Description

Values is an array of strings, which is compared against the input, for guard checking It must be non-empty

Type

`array`

`.spec.steps[].when[].values[]`

Type

`string`

`.spec.steps[].workspaces`

Description

This is an alpha field. You must set the "enable-api-fields" feature flag to "alpha" for this field to be supported. Workspaces is a list of workspaces from the Task that this Step wants exclusive access to. Adding a workspace to this list means that any other Step or Sidecar that does not also request this Workspace will not have access to it.

Type

`array`

`.spec.steps[].workspaces[]`

Description

WorkspaceUsage is used by a Step or Sidecar to declare that it wants isolated access to a Workspace defined in a Task.

Type

`object`

Required

`mountPath``name`

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath is the path that the workspace should be mounted to inside the Step or Sidecar, overriding any MountPath specified in the Task's WorkspaceDeclaration.
<code>name</code>	<code>string</code>	Name is the name of the workspace this Step or Sidecar wants access to.

`.spec.volumes`

Description

Volumes is a collection of volumes that are available to mount into the steps of the build.

Type

`array`

`.spec.volumes[]`

Description

Volume represents a named volume in a pod that may be accessed by any container in the pod.

Type

`object`

Required

`name`

Property	Type	Description
<code>awsElasticBlockStore</code>	<code>object</code>	<code>awsElasticBlockStore</code> represents an AWS Disk resource a kubelet's host machine and then exposed to the pod.

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/volumes#azureDisk 
<code>azureDisk</code>	<code>object</code>	azureDisk represents an Azure Data Disk mount on the pod.
<code>azureFile</code>	<code>object</code>	azureFile represents an Azure File Service mount on the pod.
<code>cephfs</code>	<code>object</code>	cephFS represents a Ceph FS mount on the host that shares a pod's lifetime
<code>cinder</code>	<code>object</code>	cinder represents a cinder volume attached and mounted on the machine. More info: https://examples.k8s.io/mysql-cinder-pd/README.md 
<code>configMap</code>	<code>object</code>	configMap represents a configMap that should populate this volume
<code>csi</code>	<code>object</code>	csi (Container Storage Interface) represents ephemeral storage handled by certain external CSI drivers (Beta feature).
<code>downwardAPI</code>	<code>object</code>	downwardAPI represents downward API about the pod that should populate this volume

Property	Type	Description
<code>emptyDir</code>	<code>object</code>	<code>emptyDir</code> represents a temporary directory that shares More info: https://kubernetes.io/docs/concepts/storage/
<code>ephemeral</code>	<code>object</code>	<code>ephemeral</code> represents a volume that is handled by a cl The volume's lifecycle is tied to the pod that defines it - before the pod starts, and deleted when the pod is rem Use this if: a) the volume is only needed while the pod i normal volumes like restoring from snapshot or capacit needed, c) the storage driver is specified through a stor the storage driver supports dynamic volume provisionir PersistentVolumeClaim (see EphemeralVolumeSource information on the connection between this volume typ PersistentVolumeClaim). Use PersistentVolumeClaim or one of the vendor-speci volumes that persist for longer than the lifecycle of an i Use CSI for light-weight local ephemeral volumes if the meant to be used that way - see the documentation of t information. A pod can use both types of ephemeral volumes and pr at the same time.
<code>fc</code>	<code>object</code>	<code>fc</code> represents a Fibre Channel resource that is attached machine and then exposed to the pod.

Property	Type	Description
<code>flexVolume</code>	<code>object</code>	<code>flexVolume</code> represents a generic volume resource that is provisioned/attached using an exec based plugin.
<code>flocker</code>	<code>object</code>	<code>flocker</code> represents a Flocker volume attached to a kubelet. This depends on the Flocker control service being run.
<code>gcePersistentDisk</code>	<code>object</code>	<code>gcePersistentDisk</code> represents a GCE Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes#gce-persistent-disk
<code>gitRepo</code>	<code>object</code>	<code>gitRepo</code> represents a git repository at a particular revision. <code>GitRepo</code> is deprecated. To provision a container with a git repo, use an <code>InitContainer</code> that clones the repo using <code>git</code> , and then mount the <code>EmptyDir</code> into the Pod's container.
<code>glusterfs</code>	<code>object</code>	<code>glusterfs</code> represents a Glusterfs mount on the host that is exposed to the container. More info: https://examples.k8s.io/volumes/glusterfs/
<code>hostPath</code>	<code>object</code>	<code>hostPath</code> represents a pre-existing file or directory on the host that is directly exposed to the container. This is general purpose and can be used to expose agents or other privileged things that are allowed to see the host's file system. Most containers will NOT need this. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath

Property	Type	Description
iscsi	object	iscsi represents an ISCSI Disk resource that is attached to the host machine and then exposed to the pod. More info: https://examples.k8s.io/volumes/iscsi/README.md ↗
name	string	name of the volume. Must be a DNS_LABEL and unique within the namespace. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
nfs	object	nfs represents an NFS mount on the host that shares a folder under a path exported by a NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes/#nfs
persistentVolumeClaim	object	persistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes/#persistentvolumeclaims ↗
photonPersistentDisk	object	photonPersistentDisk represents a PhotonController persistent disk attached and mounted on kubelets host machine
portworxVolume	object	portworxVolume represents a portworx volume attached to a kubelet's host machine
projected	object	projected items for all in one resources secrets, configmaps, downward API

Property	Type	Description
<code>quobyte</code>	<code>object</code>	quobyte represents a Quobyte mount on the host that s lifetime
<code>rbd</code>	<code>object</code>	rbd represents a Rados Block Device mount on the hos pod's lifetime. More info: https://examples.k8s.io/volumes/rbd/README.md ↗
<code>scaleIO</code>	<code>object</code>	scaleIO represents a ScaleIO persistent volume attach Kubernetes nodes.
<code>secret</code>	<code>object</code>	secret represents a secret that should populate this vol https://kubernetes.io/docs/concepts/storage/volumes#s
<code>storageos</code>	<code>object</code>	storageOS represents a StorageOS volume attached a Kubernetes nodes.
<code>vsphereVolume</code>	<code>object</code>	vsphereVolume represents a vSphere volume attached kubelets host machine

`.spec.volumes[].awsElasticBlockStore`

Description

awsElasticBlockStore represents an AWS Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore>

Type

object

Required

volumeID

Property	Type	Description
fsType	string	fsType is the filesystem type of the volume that you want to mount. 1 Ensure that the filesystem type is supported by the host operating sy Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspe More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblo
partition	integer	partition is the partition in the volume that you want to mount. If omit the default is to mount by volume name. Examples: For volume /dev you specify the partition as "1". Similarly, the volume partition for /de is "0" (or you can leave the property empty).
readOnly	boolean	readOnly value true will force the readOnly setting in VolumeMounts info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblo
volumeID	string	volumeID is unique ID of the persistent disk resource in AWS (Amaz EBS volume). More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblo

.spec.volumes[].azureDisk

Description

azureDisk represents an Azure Data Disk mount on the host and bind mount to the pod.

Type

object

Required

diskName

diskURI

Property	Type	Description
<code>cachedMode</code>	<code>string</code>	cachedMode is the Host Caching mode: None, Read Only, Read Write.
<code>diskName</code>	<code>string</code>	diskName is the Name of the data disk in the blob storage
<code>diskURI</code>	<code>string</code>	diskURI is the URI of data disk in the blob storage
<code>fsType</code>	<code>string</code>	fsType is Filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
<code>kind</code>	<code>string</code>	kind expected values are Shared: multiple blob disks per storage account Dedicated: single blob disk per storage account Managed: azure managed data disk (only in managed availability set). defaults to shared

Property	Type	Description
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> Defaults to false (read/write). <code>ReadOnly</code> here will force the <code>ReadOnly</code> setting in <code>VolumeMounts</code> .

`.spec.volumes[].azureFile`

Description

`azureFile` represents an Azure File Service mount on the host and bind mount to the pod.

Type

`object`

Required

`secretName`

`shareName`

Property	Type	Description
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> defaults to false (read/write). <code>ReadOnly</code> here will force the <code>ReadOnly</code> setting in <code>VolumeMounts</code> .
<code>secretName</code>	<code>string</code>	<code>secretName</code> is the name of secret that contains Azure Storage Account Name and Key
<code>shareName</code>	<code>string</code>	<code>shareName</code> is the azure share Name

`.spec.volumes[].cephfs`

Description

`cephFS` represents a Ceph FS mount on the host that shares a pod's lifetime

Type

object

Required

monitors

Property	Type	Description
<code>monitors</code>	<code>array</code>	<code>monitors</code> is Required: Monitors is a collection of Ceph monitors More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it ↗
<code>path</code>	<code>string</code>	<code>path</code> is Optional: Used as the mounted root, rather than the full Ceph tree, default is /
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> is Optional: Defaults to false (read/write). <code>ReadOnly</code> here will force the <code>ReadOnly</code> setting in <code>VolumeMounts</code>. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it ↗
<code>secretFile</code>	<code>string</code>	<code>secretFile</code> is Optional: <code>SecretFile</code> is the path to key ring for User, default is <code>/etc/ceph/user.secret</code> More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it ↗

Property	Type	Description
<code>secretRef</code>	<code>object</code>	secretRef is Optional: SecretRef is reference to the authentication secret for User, default is empty. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it ↗
<code>user</code>	<code>string</code>	user is optional: User is the rados user name, default is admin More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it ↗

`.spec.volumes[].cephfs.monitors`

Description

monitors is Required: Monitors is a collection of Ceph monitors More info: <https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it>

Type

`array`

`.spec.volumes[].cephfs.monitors[]`

Type

`string`

`.spec.volumes[].cephfs.secretRef`

Description

secretRef is Optional: SecretRef is reference to the authentication secret for User, default is empty. More info: <https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it>

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

`.spec.volumes[].cinder`

Description

cinder represents a cinder volume attached and mounted on kubelets host machine. More info: <https://examples.k8s.io/mysql-cinder-pd/README.md>

Type

`object`

Required

`volumeID`

Property	Type	Description
<code>fsType</code>	<code>string</code>	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://examples.k8s.io/mysql-cinder-pd/README.md ↗
<code>readOnly</code>	<code>boolean</code>	readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/mysql-cinder-pd/README.md ↗

Property	Type	Description
<code>secretRef</code>	<code>object</code>	secretRef is optional: points to a secret object containing parameters used to connect to OpenStack.
<code>volumeID</code>	<code>string</code>	volumeID used to identify the volume in cinder. More info: https://examples.k8s.io/mysql-cinder-pd/README.md ↗

`.spec.volumes[].cinder.secretRef`

Description

secretRef is optional: points to a secret object containing parameters used to connect to OpenStack.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

`.spec.volumes[].configMap`

Description

configMap represents a configMap that should populate this volume

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> is optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code>, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	<code>items</code> if unspecified, each key-value pair in the <code>Data</code> field of the referenced <code>ConfigMap</code> will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the <code>ConfigMap</code>, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the <code>'..'</code> path or start with <code>'..'</code>.
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	optional specify whether the <code>ConfigMap</code> or its keys must be defined

`.spec.volumes[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.volumes[].configMap.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

Property	Type	Description
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.volumes[].csi`

Description

csi (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers (Beta feature).

Type

`object`

Required

`driver`

Property	Type	Description
<code>driver</code>	<code>string</code>	driver is the name of the CSI driver that handles this volume. Consult with your admin for the correct name as registered in the cluster.
<code>fsType</code>	<code>string</code>	fsType to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.
<code>nodePublishSecretRef</code>	<code>object</code>	nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI

Property	Type	Description
		NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> specifies a read-only configuration for the volume. Defaults to false (read/write).
<code>volumeAttributes</code>	<code>object</code>	<code>volumeAttributes</code> stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

`.spec.volumes[].csi.nodePublishSecretRef`

Description

`nodePublishSecretRef` is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI `NodePublishVolume` and `NodeUnpublishVolume` calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

`.spec.volumes[].csi.volumeAttributes`

Description

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

Type

object

`.spec.volumes[].downwardAPI`

Description

downwardAPI represents downward API about the pod that should populate this volume

Type

object

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	Optional: mode bits to use on created files by default. Must be a Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	Items is a list of downward API volume file

`.spec.volumes[].downwardAPI.items`

Description

Items is a list of downward API volume file

Type

array

.spec.volumes[].downwardAPI.items[]

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

object

Required

path

Property	Type	Description
fieldRef	object	Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.
mode	integer	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..'

Property	Type	Description
		path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

`.spec.volumes[].downwardAPI.items[].fieldRef`

Description

Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.volumes[].downwardAPI.items[].resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

.spec.volumes[].emptyDir

Description

emptyDir represents a temporary directory that shares a pod's lifetime. More info: <https://kubernetes.io/docs/concepts/storage/volumes#emptydir>

Type

object

Property	Type	Description
<code>medium</code>	<code>string</code>	medium represents what type of storage medium should back this directory. The default is "" which means to use the node's default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir
<code>sizeLimit</code>		sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

`.spec.volumes[].ephemeral`

Description

ephemeral represents a volume that is handled by a cluster storage driver. The volume's lifecycle is tied to the pod that defines it - it will be created before the pod starts, and deleted when the pod is removed. Use this if: a) the volume is only needed while the pod runs, b) features of normal volumes like restoring from snapshot or capacity tracking are needed, c) the storage driver is specified through a storage class, and d) the storage driver supports dynamic volume provisioning through a PersistentVolumeClaim (see EphemeralVolumeSource for more information on the connection between this volume type and PersistentVolumeClaim). Use PersistentVolumeClaim or one of the vendor-specific APIs for volumes that persist for longer than the lifecycle of an individual pod. Use CSI for light-weight local ephemeral volumes if the CSI driver is meant to be used that way - see

the documentation of the driver for more information. A pod can use both types of ephemeral volumes and persistent volumes at the same time.

Type

object

Property	Type	Description
		Will be used to create a stand-alone PVC to provision the volume. The pod in which this EphemeralVolumeSource is embedded will be the owner of the PVC, i.e. the PVC will be deleted together with the pod. The name of the PVC will be <code><pod name>-<volume name></code> where <code><volume name></code> is the name from the <code>PodSpec.Volumes</code> array entry. Pod validation will reject the pod if the concatenated name is not valid for a PVC (for example, too long).
<code>volumeClaimTemplate</code>	object	An existing PVC with that name that is not owned by the pod will <i>not</i> be used for the pod to avoid using an unrelated volume by mistake. Starting the pod is then blocked until the unrelated PVC is removed. If such a pre-created PVC is meant to be used by the pod, the PVC has to updated with an owner reference to the pod once the pod exists. Normally this should not be necessary, but it may be useful when manually reconstructing a broken cluster. This field is read-only and no changes will be made by Kubernetes to the PVC after it has been created. Required, must not be nil.

`.spec.volumes[].ephemeral.volumeClaimTemplate`

Description

Will be used to create a stand-alone PVC to provision the volume. The pod in which this EphemeralVolumeSource is embedded will be the owner of the PVC, i.e. the PVC will be deleted together with the pod. The name of the PVC will be ``<pod name>-<volume name>`` where ``<volume name>`` is the name from the ``PodSpec.Volumes`` array entry. Pod validation will reject the pod if the concatenated name is not valid for a PVC (for example, too long). An existing PVC with that name that is not owned by the pod will *not* be used for the pod to avoid using an unrelated volume by mistake. Starting the pod is then blocked until the unrelated PVC is removed. If such a pre-created PVC is meant to be used by the pod, the PVC has to be updated with an owner reference to the pod once the pod exists. Normally this should not be necessary, but it may be useful when manually reconstructing a broken cluster. This field is read-only and no changes will be made by Kubernetes to the PVC after it has been created. Required, must not be nil.

Type

object

Required

spec

Property	Type	Description
metadata	ObjectMeta	May contain labels and annotations that will be copied into the PVC when creating it. No other fields are allowed and will be rejected during validation.
spec	object	The specification for the PersistentVolumeClaim. The entire content is copied unchanged into the PVC that gets created from this template. The same fields as in a PersistentVolumeClaim are also valid here.

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec`

Description

The specification for the PersistentVolumeClaim. The entire content is copied unchanged into the PVC that gets created from this template. The same fields as in a PersistentVolumeClaim are also valid here.

Type

object

Property	Type	Description
accessModes	array	accessModes contains the desired access modes volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistentvolumes#access-modes-1
dataSource	object	dataSource field can be used to specify either: - An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) - An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.
dataSourceRef	object	dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a

Property	Type	Description
		PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: • While dataSource only allows two specific type objects, dataSourceRef allows any non-core object as well as PersistentVolumeClaim objects. • While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. • While dataSource only allows local objects, dataSourceRef allows objects in any namespace (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.
resources	object	resources represents the minimum resources the volume should have. If RecoverVolumeExpansionFailure feature is enabled

Property	Type	Description
		users are allowed to specify resource requirement that are lower than previous value but must still be higher than capacity recorded in the status field of claim. More info: https://kubernetes.io/docs/concepts/storage/persistentvolumes#resources ↗
<code>selector</code>	<code>object</code>	selector is a label query over volumes to consider binding.
<code>storageClassName</code>	<code>string</code>	storageClassName is the name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistentvolumes#class-1 ↗
<code>volumeAttributesClassName</code>	<code>string</code>	volumeAttributesClassName may be used to set the VolumeAttributesClass used by this claim. If specified the CSI driver will create or update the volume with attributes defined in the corresponding VolumeAttributesClass. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string value means that VolumeAttributesClass will be applied to the claim it's not allowed to reset this field to empty string once it is set. If unspecified and the PersistentVolumeClaim is unbound, the default VolumeAttributesClass will be used by the persistentvolume controller if it exists. If the resource referred to by volumeAttributesClassName does not exist, this PersistentVolumeClaim will be set to Pending state, as reflected by the modifyVolumeStatus field, until such as a resource exists. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/persistent-volumes#volumeattributesclass [↗] (Alpha) Using this field requires the VolumeAttributesClass feature gate to be enabled.
<code>volumeMode</code>	<code>string</code>	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.
<code>volumeName</code>	<code>string</code>	volumeName is the binding reference to the PersistentVolume backing this claim.

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.accessModes`

Description

accessModes contains the desired access modes the volume should have. More info: <https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1>

Type

`array`

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.accessModes[]`

Type

`string`

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.dataSource`

Description

dataSource field can be used to specify either: * An existing VolumeSnapshot object (snapshot.storage.k8s.io/VolumeSnapshot) * An existing PVC (PersistentVolumeClaim) If the provisioner or an external controller can support the specified data source, it will create a new volume based on the contents of the specified data source. When the AnyVolumeDataSource feature gate is enabled, dataSource contents will be copied to dataSourceRef, and dataSourceRef contents will be copied to dataSource when dataSourceRef.namespace is not specified. If the namespace is specified, then dataSourceRef will not be copied to dataSource.

Type

`object`

Required

`kind``name`

Property	Type	Description
<code>apiGroup</code>	<code>string</code>	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
<code>kind</code>	<code>string</code>	Kind is the type of resource being referenced
<code>name</code>	<code>string</code>	Name is the name of resource being referenced

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.dataSourceRef`

Description

`dataSourceRef` specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a `PersistentVolumeClaim` object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the `dataSource` field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in `dataSourceRef`, both fields (`dataSource` and `dataSourceRef`) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in `dataSourceRef`, `dataSource` isn't set to the same value and must be empty. There are three important differences between `dataSource` and `dataSourceRef`:

- * While `dataSource` only allows two specific types of objects, `dataSourceRef` allows any non-core object, as well as `PersistentVolumeClaim` objects.
- * While `dataSource` ignores disallowed values (dropping them), `dataSourceRef` preserves all values, and generates an error if a disallowed value is specified.
- * While `dataSource` only allows local objects, `dataSourceRef` allows objects in any namespaces.

(Beta) Using this field requires the `AnyVolumeDataSource` feature gate to be enabled.
(Alpha) Using the namespace field of `dataSourceRef` requires the `CrossNamespaceVolumeDataSource` feature gate to be enabled.

Type

object

Required

kind

name

Property	Type	Description
<code>apiGroup</code>	<code>string</code>	APIGroup is the group for the resource being referenced. If APIGroup is not specified, the specified Kind must be in the core API group. For any other third-party types, APIGroup is required.
<code>kind</code>	<code>string</code>	Kind is the type of resource being referenced

Property	Type	Description
<code>name</code>	<code>string</code>	Name is the name of resource being referenced
<code>namespace</code>	<code>string</code>	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a <code>gateway.networking.k8s.io/ReferenceGrant</code> object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the <code>CrossNamespaceVolumeDataSource</code> feature gate to be enabled.

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.resources`

Description

`resources` represents the minimum resources the volume should have. If `RecoverVolumeExpansionFailure` feature is enabled users are allowed to specify resource requirements that are lower than previous value but must still be higher than capacity recorded in the status field of the claim. More info: <https://kubernetes.io/docs/concepts/storage/persistent-volumes#resources>

Type

`object`

Property	Type	Description
<code>limits</code>	<code>object</code>	Limits describes the maximum amount of compute resources allowed. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗
<code>requests</code>	<code>object</code>	Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an implementation-defined value. Requests cannot exceed Limits. More info: https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.resources.limits`

Description

Limits describes the maximum amount of compute resources allowed. More info: <https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

`object`

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.resources.requests`

Description

Requests describes the minimum amount of compute resources required. If Requests is omitted for a container, it defaults to Limits if that is explicitly specified, otherwise to an

implementation-defined value. Requests cannot exceed Limits. More info:
<https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/>

Type

object

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.selector`

Description

selector is a label query over volumes to consider for binding.

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	object	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.selector.matchExpressions`

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.selector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

key

operator

Property	Type	Description
key	string	key is the label key that the selector applies to.
operator	string	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	array	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.volumes[].ephemeral.volumeClaimTemplate.spec.selector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

array

.spec.volumes[].ephemeral.volumeClaimTemplate.spec.selector.matchExpressions[].values[]

Type

string

.spec.volumes[].ephemeral.volumeClaimTemplate.spec.selector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

.spec.volumes[].fc

Description

fc represents a Fibre Channel resource that is attached to a kubelet's host machine and then exposed to the pod.

Type

object

Property	Type	Description
<code>fsType</code>	<code>string</code>	<code>fsType</code> is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
<code>lun</code>	<code>integer</code>	<code>lun</code> is Optional: FC target lun number
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> is Optional: Defaults to false (read/write). <code>ReadOnly</code> here will force the <code>ReadOnly</code> setting in <code>VolumeMounts</code> .
<code>targetWWNs</code>	<code>array</code>	<code>targetWWNs</code> is Optional: FC target worldwide names (WWNs)
<code>wwids</code>	<code>array</code>	<code>wwids</code> Optional: FC volume world wide identifiers (wwids) Either <code>wwids</code> or combination of <code>targetWWNs</code> and <code>lun</code> must be set, but not both simultaneously.

`.spec.volumes[].fc.targetWWNs`

Description

`targetWWNs` is Optional: FC target worldwide names (WWNs)

Type

`array`

`.spec.volumes[].fc.targetWWNs[]`

Type

`string`

`.spec.volumes[].fc.wwids`

Description

wwids Optional: FC volume world wide identifiers (wwids) Either wwids or combination of targetWWNs and lun must be set, but not both simultaneously.

Type

`array`

`.spec.volumes[].fc.wwids[]`

Type

`string`

`.spec.volumes[].flexVolume`

Description

flexVolume represents a generic volume resource that is provisioned/attached using an exec based plugin.

Type

`object`

Required

`driver`

Property	Type	Description
<code>driver</code>	<code>string</code>	driver is the name of the driver to use for this volume.
<code>fsType</code>	<code>string</code>	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs",

Property	Type	Description
		"ntfs". The default filesystem depends on FlexVolume script.
<code>options</code>	<code>object</code>	options is Optional: this field holds extra command options if any.
<code>readOnly</code>	<code>boolean</code>	readOnly is Optional: defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
<code>secretRef</code>	<code>object</code>	secretRef is Optional: secretRef is reference to the secret object containing sensitive information to pass to the plugin scripts. This may be empty if no secret object is specified. If the secret object contains more than one secret, all secrets are passed to the plugin scripts.

`.spec.volumes[].flexVolume.options`

Description

options is Optional: this field holds extra command options if any.

Type

`object`

`.spec.volumes[].flexVolume.secretRef`

Description

secretRef is Optional: secretRef is reference to the secret object containing sensitive information to pass to the plugin scripts. This may be empty if no secret object is specified. If the secret object contains more than one secret, all secrets are passed to the plugin scripts.

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].flocker

Description

flocker represents a Flocker volume attached to a kubelet's host machine. This depends on the Flocker control service being running

Type

object

Property	Type	Description
<code>datasetName</code>	<code>string</code>	datasetName is Name of the dataset stored as metadata -> name on the dataset for Flocker should be considered as deprecated
<code>datasetUUID</code>	<code>string</code>	datasetUUID is the UUID of the dataset. This is unique identifier of a Flocker dataset

.spec.volumes[].gcePersistentDisk

Description

`gcePersistentDisk` represents a GCE Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk>

Type

object

Required

pdName

Property	Type	Description
<code>fsType</code>	<code>string</code>	<code>fsType</code> is filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk
<code>partition</code>	<code>integer</code>	<code>partition</code> is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume <code>/dev/sda1</code>, you specify the partition as "1". Similarly, the volume partition for <code>/dev/sda</code> is "0" (or you can leave the property empty). More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk
<code>pdName</code>	<code>string</code>	<code>pdName</code> is unique name of the PD resource in GCE. Used to identify the disk in GCE. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk

Property	Type	Description
<code>readOnly</code>	<code>boolean</code>	readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentvolumeclaim

`.spec.volumes[].gitRepo`

Description

gitRepo represents a git repository at a particular revision. DEPRECATED: GitRepo is deprecated. To provision a container with a git repo, mount an EmptyDir into an InitContainer that clones the repo using git, then mount the EmptyDir into the Pod's container.

Type

`object`

Required

`repository`

Property	Type	Description
<code>directory</code>	<code>string</code>	directory is the target directory name. Must not contain or start with '..'. If '.' is supplied, the volume directory will be the git repository. Otherwise, if specified, the volume will contain the git repository in the subdirectory with the given name.
<code>repository</code>	<code>string</code>	repository is the URL
<code>revision</code>	<code>string</code>	revision is the commit hash for the specified revision.

`.spec.volumes[].glusterfs`

Description

glusterfs represents a Glusterfs mount on the host that shares a pod's lifetime. More info: <https://examples.k8s.io/volumes/glusterfs/README.md>

Type

object

Required

endpoints

path

Property	Type	Description
<code>endpoints</code>	<code>string</code>	endpoints is the endpoint name that details Glusterfs topology. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod
<code>path</code>	<code>string</code>	path is the Glusterfs volume path. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod
<code>readOnly</code>	<code>boolean</code>	readOnly here will force the Glusterfs volume to be mounted with read-only permissions. Defaults to false. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod

`.spec.volumes[].hostPath`

Description

hostPath represents a pre-existing file or directory on the host machine that is directly exposed to the container. This is generally used for system agents or other privileged things that are allowed to see the host machine. Most containers will NOT need this. More info: <https://kubernetes.io/docs/concepts/storage/volumes#hostpath>

Type

object

Required

path

Property	Type	Description
path	string	path of the directory on the host. If the path is a symlink, it will follow the link to the real path. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath ↗
type	string	type for HostPath Volume Defaults to "" More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath ↗

.spec.volumes[].iscsi

Description

iscsi represents an ISCSI Disk resource that is attached to a kubelet's host machine and then exposed to the pod. More info: <https://examples.k8s.io/volumes/iscsi/README.md>

Type

object

Required

iqn

lun

targetPortal

Property	Type	Description
<code>chapAuthDiscovery</code>	<code>boolean</code>	chapAuthDiscovery defines whether support iSCSI Discovery CHAP authentication
<code>chapAuthSession</code>	<code>boolean</code>	chapAuthSession defines whether support iSCSI Session CHAP authentication
<code>fsType</code>	<code>string</code>	fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#iscsi
<code>initiatorName</code>	<code>string</code>	initiatorName is the custom iSCSI Initiator Name. If initiatorName is specified with iscsiInterface simultaneously, new iSCSI interface : will be created for the connection.
<code>iqn</code>	<code>string</code>	iqn is the target iSCSI Qualified Name.
<code>iscsiInterface</code>	<code>string</code>	iscsiInterface is the interface Name that uses an iSCSI transport. Defaults to 'default' (tcp).
<code>lun</code>	<code>integer</code>	lun represents iSCSI Target Lun number.

Property	Type	Description
<code>portals</code>	<code>array</code>	portals is the iSCSI Target Portal List. The portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).
<code>readOnly</code>	<code>boolean</code>	readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false.
<code>secretRef</code>	<code>object</code>	secretRef is the CHAP Secret for iSCSI target and initiator authentication
<code>targetPortal</code>	<code>string</code>	targetPortal is iSCSI Target Portal. The Portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

`.spec.volumes[].iscsi.portals`

Description

portals is the iSCSI Target Portal List. The portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

Type

`array`

`.spec.volumes[].iscsi.portals[]`

Type

`string`

`.spec.volumes[].iscsi.secretRef`

Description

secretRef is the CHAP Secret for iSCSI target and initiator authentication

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

`.spec.volumes[].nfs`

Description

nfs represents an NFS mount on the host that shares a pod's lifetime More info:
<https://kubernetes.io/docs/concepts/storage/volumes#nfs>

Type

object

Required

`path` `server`

Property	Type	Description
<code>path</code>	<code>string</code>	path that is exported by the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗
<code>readOnly</code>	<code>boolean</code>	readOnly here will force the NFS export to be mounted with read-only permissions. Defaults to false. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗
<code>server</code>	<code>string</code>	server is the hostname or IP address of the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs ↗

`.spec.volumes[].persistentVolumeClaim`

Description

`persistentVolumeClaimVolumeSource` represents a reference to a `PersistentVolumeClaim` in the same namespace. More info: <https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims>

Type

`object`

Required

`claimName`

Property	Type	Description
<code>claimName</code>	<code>string</code>	<code>claimName</code> is the name of a <code>PersistentVolumeClaim</code> in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> Will force the <code>ReadOnly</code> setting in <code>VolumeMounts</code> . Default false.

`.spec.volumes[].photonPersistentDisk`

Description

photonPersistentDisk represents a PhotonController persistent disk attached and mounted on kubelets host machine

Type

object

Required

pdID

Property	Type	Description
fsType	string	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
pdID	string	pdID is the ID that identifies Photon Controller persistent disk

.spec.volumes[].portworxVolume

Description

portworxVolume represents a portworx volume attached and mounted on kubelets host machine

Type

object

Required

volumeID

Property	Type	Description
fsType	string	fsType represents the filesystem type to mount Must be a filesystem type supported by the host operating system. Ex.

Property	Type	Description
		"ext4", "xfs". Implicitly inferred to be "ext4" if unspecified.
<code>readOnly</code>	<code>boolean</code>	readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
<code>volumeID</code>	<code>string</code>	volumeID uniquely identifies a Portworx volume

`.spec.volumes[].projected`

Description

projected items for all in one resources secrets, configmaps, and downward API

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	defaultMode are the mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>sources</code>	<code>array</code>	sources is the list of volume projections

`.spec.volumes[].projected.sources`

Description

`sources` is the list of volume projections

Type

array

`.spec.volumes[].projected.sources[]`

Description

Projection that may be projected along with other supported volume types

Type

object

Property	Type	Description
<code>clusterTrustBundle</code>	object	ClusterTrustBundle allows a pod to access the <code>.spec.trustBundle</code> field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.

Property	Type	Description
<code>configMap</code>	<code>object</code>	configMap information about the configMap data to project
<code>downwardAPI</code>	<code>object</code>	downwardAPI information about the downwardAPI data to project
<code>secret</code>	<code>object</code>	secret information about the secret data to project
<code>serviceAccountToken</code>	<code>object</code>	serviceAccountToken is information about the serviceAccountToken data to project

`.spec.volumes[].projected.sources[].clusterTrustBundle`

Description

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.`

Type

`object`

Required

`path`

Property	Type	Description
<code>labelSelector</code>	<code>object</code>	Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".
<code>name</code>	<code>string</code>	Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.
<code>optional</code>	<code>boolean</code>	If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.
<code>path</code>	<code>string</code>	Relative path from the volume root to write the bundle.
<code>signerName</code>	<code>string</code>	Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all selected ClusterTrustBundles will be unified and deduplicated.

`.spec.volumes[].projected.sources[].clusterTrustBundle.labelSelector`

Description

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	object	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.volumes[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions`

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

`.spec.volumes[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

key

operator

Property	Type	Description
key	string	key is the label key that the selector applies to.
operator	string	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	array	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.volumes[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

array

`.spec.volumes[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values[]`

Type

string

`.spec.volumes[].projected.sources[].clusterTrustBundle.labelSelector.matchLabels`

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

`.spec.volumes[].projected.sources[].configMap`

Description

configMap information about the configMap data to project

Type

object

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Property	Type	Description
name	string	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	optional specify whether the ConfigMap or its keys must be defined

`.spec.volumes[].projected.sources[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

`.spec.volumes[].projected.sources[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

object

Required

key path

Property	Type	Description
<code>key</code>	<code>string</code>	key is the key to project.
<code>mode</code>	<code>integer</code>	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.volumes[].projected.sources[].downwardAPI`

Description

downwardAPI information about the downwardAPI data to project

Type

`object`

Property	Type	Description
<code>items</code>	<code>array</code>	Items is a list of DownwardAPIVolume file

`.spec.volumes[].projected.sources[].downwardAPI.items`

Description

Items is a list of DownwardAPIVolume file

Type

array

`.spec.volumes[].projected.sources[].downwardAPI.items[]`

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

object

Required

path

Property	Type	Description
<code>fieldRef</code>	<code>object</code>	Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.
<code>mode</code>	<code>integer</code>	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..'

Property	Type	Description
		path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

`.spec.volumes[].projected.sources[].downwardAPI.items[].fieldRef`

Description

Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.volumes[].projected.sources[].downwardAPI.items[].resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>	<code>int</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.volumes[].projected.sources[].secret`

Description

secret information about the secret data to project

Type

`object`

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its key must be defined

`.spec.volumes[].projected.sources[].secret.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.volumes[].projected.sources[].secret.items[]`

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.volumes[].projected.sources[].serviceAccountToken`

Description

serviceAccountToken is information about the serviceAccountToken data to project

Type

`object`

Required

`path`

Property	Type	Description
<code>audience</code>	<code>string</code>	audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.
<code>expirationSeconds</code>	<code>integer</code>	expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token. The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.
<code>path</code>	<code>string</code>	path is the path relative to the mount point of the file to project the token into.

`.spec.volumes[].quobyte`

Description

quobyte represents a Quobyte mount on the host that shares a pod's lifetime

Type

`object`

Required

registry

volume

Property	Type	Description
group	string	group to map volume access to Default is no group
readOnly	boolean	readOnly here will force the Quobyte volume to be mounted with read-only permissions. Defaults to false.
registry	string	registry represents a single or multiple Quobyte Registry services specified as a string as host:port pair (multiple entries are separated with commas) which acts as the central registry for volumes
tenant	string	tenant owning the given Quobyte volume in the Backend Used with dynamically provisioned Quobyte volumes, value is set by the plugin
user	string	user to map volume access to Defaults to serviceaccount user
volume	string	volume is a string that references an already created Quobyte volume by name.

.spec.volumes[].rbd

Description

rbid represents a Rados Block Device mount on the host that shares a pod's lifetime. More info: <https://examples.k8s.io/volumes/rbd/README.md>

Type

object

Required

image

monitors

Property	Type	Description
fsType	string	fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#rbd
image	string	image is the rados image name. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
keyring	string	keyring is the path to key ring for RBDUser. Default is /etc/ceph/keyring. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
monitors	array	monitors is a collection of Ceph monitors. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

Property	Type	Description
<code>pool</code>	<code>string</code>	pool is the rados pool name. Default is rbd. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
<code>readOnly</code>	<code>boolean</code>	readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
<code>secretRef</code>	<code>object</code>	secretRef is name of the authentication secret for RBDUser. If provided overrides keyring. Default is nil. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
<code>user</code>	<code>string</code>	user is the rados user name. Default is admin. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

`.spec.volumes[].rbd.monitors`

Description

monitors is a collection of Ceph monitors. More info:
<https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it>

Type

`array`

`.spec.volumes[].rbd.monitors[]`

Type

string

.spec.volumes[].rbd.secretRef

Description

secretRef is name of the authentication secret for RBDUser. If provided overrides keyring. Default is nil. More info: <https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it>

Type

object

Property	Type	Description
name	string	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.volumes[].scaleIO

Description

scaleIO represents a ScaleIO persistent volume attached and mounted on Kubernetes nodes.

Type

object

Required

gateway

secretRef

system

Property	Type	Description
fsType	string	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating

Property	Type	Description
		system. Ex. "ext4", "xfs", "ntfs". Default is "xfs".
<code>gateway</code>	<code>string</code>	gateway is the host address of the ScaleIO API Gateway.
<code>protectionDomain</code>	<code>string</code>	protectionDomain is the name of the ScaleIO Protection Domain for the configured storage.
<code>readOnly</code>	<code>boolean</code>	readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
<code>secretRef</code>	<code>object</code>	secretRef references to the secret for ScaleIO user and other sensitive information. If this is not provided, Login operation will fail.
<code>sslEnabled</code>	<code>boolean</code>	sslEnabled Flag enable/disable SSL communication with Gateway, default false
<code>storageMode</code>	<code>string</code>	storageMode indicates whether the storage for a volume should be ThickProvisioned or ThinProvisioned. Default is ThinProvisioned.
<code>storagePool</code>	<code>string</code>	storagePool is the ScaleIO Storage Pool associated with the protection domain.

Property	Type	Description
<code>system</code>	<code>string</code>	system is the name of the storage system as configured in ScaleIO.
<code>volumeName</code>	<code>string</code>	volumeName is the name of a volume already created in the ScaleIO system that is associated with this volume source.

`.spec.volumes[].scaleIO.secretRef`

Description

secretRef references to the secret for ScaleIO user and other sensitive information. If this is not provided, Login operation will fail.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

`.spec.volumes[].secret`

Description

secret represents a secret that should populate this volume. More info:
<https://kubernetes.io/docs/concepts/storage/volumes#secret>

Type

object

Property	Type	Description
defaultMode	integer	defaultMode is Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
items	array	items If unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
optional	boolean	optional field specify whether the Secret or its keys must be defined
secretName	string	secretName is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

`.spec.volumes[].secret.items`

Description

items If unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

`.spec.volumes[].secret.items[]`

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
<code>key</code>	<code>string</code>	key is the key to project.

Property	Type	Description
<code>mode</code>	<code>integer</code>	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.volumes[].storageos`

Description

storageOS represents a StorageOS volume attached and mounted on Kubernetes nodes.

Type

`object`

Property	Type	Description
<code>fsType</code>	<code>string</code>	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.

Property	Type	Description
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> defaults to false (read/write). <code>ReadOnly</code> here will force the <code>ReadOnly</code> setting in <code>VolumeMounts</code> .
<code>secretRef</code>	<code>object</code>	<code>secretRef</code> specifies the secret to use for obtaining the StorageOS API credentials. If not specified, default values will be attempted.
<code>volumeName</code>	<code>string</code>	<code>volumeName</code> is the human-readable name of the StorageOS volume. Volume names are only unique within a namespace.
<code>volumeNamespace</code>	<code>string</code>	<code>volumeNamespace</code> specifies the scope of the volume within StorageOS. If no namespace is specified then the Pod's namespace will be used. This allows the Kubernetes name scoping to be mirrored within StorageOS for tighter integration. Set <code>VolumeName</code> to any name to override the default behaviour. Set to "default" if you are not using namespaces within StorageOS. Namespaces that do not pre-exist within StorageOS will be created.

`.spec.volumes[].storageos.secretRef`

Description

`secretRef` specifies the secret to use for obtaining the StorageOS API credentials. If not specified, default values will be attempted.

Type

`object`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

`.spec.volumes[].vsphereVolume`

Description

`vsphereVolume` represents a vSphere volume attached and mounted on kubelets host machine

Type

`object`

Required

`volumePath`

Property	Type	Description
<code>fsType</code>	<code>string</code>	<code>fsType</code> is filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
<code>storagePolicyID</code>	<code>string</code>	<code>storagePolicyID</code> is the storage Policy Based Management (SPBM) profile ID associated with the <code>StoragePolicyName</code> .

Property	Type	Description
<code>storagePolicyName</code>	<code>string</code>	storagePolicyName is the storage Policy Based Management (SPBM) profile name.
<code>volumePath</code>	<code>string</code>	volumePath is the path that identifies vSphere volume vmdk

`.spec.workspaces`

Description

Workspaces are the volumes that this Task requires.

Type

`array`

`.spec.workspaces[]`

Description

WorkspaceDeclaration is a declaration of a volume that a Task requires.

Type

`object`

Required

`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is an optional human readable description of this volume.

Property	Type	Description
<code>mountPath</code>	<code>string</code>	MountPath overrides the directory that the volume will be made available at.
<code>name</code>	<code>string</code>	Name is the name by which you can bind the volume at runtime.
<code>optional</code>	<code>boolean</code>	Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.
<code>readOnly</code>	<code>boolean</code>	ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1beta1/namespaces/{namespace}/clustertasks`
 - `DELETE` : delete collection of ClusterTask
 - `GET` : list objects of kind ClusterTask
 - `POST` : create a new ClusterTask
- `/apis/tekton.dev/v1beta1/namespaces/{namespace}/clustertasks/{name}`
 - `DELETE` : delete the specified ClusterTask
 - `GET` : read the specified ClusterTask

- **PATCH** : partially update the specified ClusterTask
- **PUT** : replace the specified ClusterTask

/apis/tekton.dev/v1beta1/namespaces/{namespace}/clusterTasks

HTTP method

DELETE

Description

delete collection of ClusterTask

HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ClusterTask

HTTP responses

HTTP code	Response body
200 - OK	ClusterTaskList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new ClusterTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>ClusterTask</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterTask</code> schema
201 - Created	<code>ClusterTask</code> schema

HTTP code	Response body
202 - Accepted	<code>ClusterTask</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1beta1/namespaces/{namespace}/clusterTasks/{name}

HTTP method

`DELETE`

Description

delete the specified ClusterTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> ✓ schema
202 - Accepted	<code>Status</code> ✓ schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified ClusterTask

HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterTask</code> schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ClusterTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are

Parameter	Type	Description
		present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterTask</code> schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ClusterTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a

Parameter	Type	Description
		BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>ClusterTask</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>ClusterTask</code> schema
201 - Created	<code>ClusterTask</code> schema
401 - Unauthorized	Empty



Run [tekton.dev/v1]

Description

Run represents a single execution of a Custom Task.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
kind	string	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗

Property	Type	Description
<code>metadata</code>	<code>ObjectMeta</code> ↗	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	RunSpec defines the desired state of Run
<code>status</code>	<code>object</code>	RunStatus defines the observed state of Run

.spec

Description

RunSpec defines the desired state of Run

Type

`object`

Property	Type	Description
<code>params</code>	<code>array</code>	Params is a list of Param
<code>podTemplate</code>	<code>object</code>	PodTemplate holds pod specific configuration
<code>ref</code>	<code>object</code>	TaskRef can be used to refer to a specific instance of a task.

Property	Type	Description
<code>retries</code>	<code>integer</code>	Used for propagating retries count to custom tasks
<code>serviceAccountName</code>	<code>string</code>	
<code>spec</code>	<code>object</code>	Spec is a specification of a custom task
<code>status</code>	<code>string</code>	Used for cancelling a run (and maybe more later on)
<code>statusMessage</code>	<code>string</code>	Status message for cancellation.
<code>timeout</code>	<code>string</code>	Time after which the custom-task times out. Refer Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration 
<code>workspaces</code>	<code>array</code>	Workspaces is a list of WorkspaceBindings from volumes to workspaces.

.spec.params

Description

Params is a list of Param

Type

`array`

.spec.params[]

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

.spec.podTemplate

Description

PodTemplate holds pod specific configuration

Type

object

Property	Type	Description
affinity		If specified, the pod's scheduling constraints. Pod.spec.affinity (API version: v1)
automountServiceAccountToken	boolean	AutomountServiceAccountToken indicates whether the pod should be running as this service account should have a token automatically mounted.

Property	Type	Description
<code>dnsConfig</code>	<code>object</code>	Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated configuration based on DNSPolicy.
<code>dnsPolicy</code>	<code>string</code>	Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. Parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.
<code>enableServiceLinks</code>	<code>boolean</code>	EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Defaults to true.
<code>env</code>	<code>array</code>	List of environment variables that can be provided to containers belonging to the pod.
<code>hostAliases</code>	<code>array</code>	HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.
<code>hostNetwork</code>	<code>boolean</code>	HostNetwork specifies whether the pod may use the host's network namespace.

Property	Type	Description
<code>imagePullSecrets</code>	<code>array</code>	ImagePullSecrets gives the name of the secret the pod to pull the image if specified
<code>nodeSelector</code>	<code>object</code>	NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match labels for the pod to be scheduled on that node. More info: https://kubernetes.io/docs/concepts/configuration/node-selector/
<code>priorityClassName</code>	<code>string</code>	If specified, indicates the pod's priority. "system-cluster-critical" and "system-node-critical" are two special keywords which indicate the highest priorities in the hierarchy; "system-cluster-critical" is higher than "system-node-critical". Any other value must be defined by creating a PriorityClass object with the matching name. If not specified, the pod priority will be default, which is zero if there is no default.
<code>runtimeClassName</code>	<code>string</code>	RuntimeClassName refers to a RuntimeClass object in the node.k8s.io group, which should be used to run this pod. If no RuntimeClass resource matches the pod, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is a class with an empty definition that uses the default runtime handler. More info: https://git.k8s.io/enhancements/keps/sig-node/runtime-class.md This is a beta feature as of Kubernetes v1.14.

Property	Type	Description
<code>schedulerName</code>	<code>string</code>	SchedulerName specifies the scheduler to be dispatch the Pod
<code>securityContext</code>	<code>object</code>	SecurityContext holds pod-level security attributes and common container settings. Optional: Default: nil. See type description for default values of each field. See Pod.spec.securityContext (API version: v1)
<code>tolerations</code>	<code>array</code>	If specified, the pod's tolerations.
<code>topologySpreadConstraints</code>	<code>array</code>	TopologySpreadConstraints controls how Pods spread across your cluster among failure domains as regions, zones, nodes, and other user-defined topology domains.
<code>volumes</code>	<code>array</code>	List of volumes that can be mounted by containers belonging to the pod. More info: https://kubernetes.io/docs/concepts/storage/volumes See Pod.spec.volumes (API version: v1)

`.spec.podTemplate.dnsConfig`

Description

Specifies the DNS parameters of a pod. Parameters specified here will be merged to the generated DNS configuration based on DNSPolicy.

Type

`object`

Property	Type	Description
<code>nameservers</code>	<code>array</code>	A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.
<code>options</code>	<code>array</code>	A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.
<code>searches</code>	<code>array</code>	A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

`.spec.podTemplate.dnsConfig.nameservers`

Description

A list of DNS name server IP addresses. This will be appended to the base nameservers generated from DNSPolicy. Duplicated nameservers will be removed.

Type

`array`

`.spec.podTemplate.dnsConfig.nameservers[]`

Type

`string`

`.spec.podTemplate.dnsConfig.options`

Description

A list of DNS resolver options. This will be merged with the base options generated from DNSPolicy. Duplicated entries will be removed. Resolution options given in Options will override those that appear in the base DNSPolicy.

Type

array

`.spec.podTemplate.dnsConfig.options[]`

Description

PodDNSConfigOption defines DNS resolver options of a pod.

Type

object

Property	Type	Description
<code>name</code>	<code>string</code>	Name is this DNS resolver option's name. Required.
<code>value</code>	<code>string</code>	Value is this DNS resolver option's value.

`.spec.podTemplate.dnsConfig.searches`

Description

A list of DNS search domains for host-name lookup. This will be appended to the base search paths generated from DNSPolicy. Duplicated search paths will be removed.

Type

array

`.spec.podTemplate.dnsConfig.searches[]`

Type

`string`

`.spec.podTemplate.env`

Description

List of environment variables that can be provided to the containers belonging to the pod.

Type

`array`

`.spec.podTemplate.env[]`

Description

EnvVar represents an environment variable present in a Container.

Type

`object`

Required

`name`

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the environment variable. Must be a C_IDENTIFIER.
<code>value</code>	<code>string</code>	Variable references \$(VAR_NAME) are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce

Property	Type	Description
		the string literal "\${VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to "".
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

.spec.podTemplate.env[].valueFrom

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports metadata.name, metadata.namespace, <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu,

Property	Type	Description
		requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.podTemplate.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

`.spec.podTemplate.env[].valueFrom.fieldRef`

Description

Selects a field of the pod: supports `metadata.name`, `metadata.namespace`, ``metadata.labels['<KEY>']``, ``metadata.annotations['<KEY>']``, `spec.nodeName`, `spec.serviceAccountName`, `status.hostIP`, `status.podIP`, `status.podIPs`.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.podTemplate.env[].valueFrom.resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (`limits.cpu`, `limits.memory`, `limits.ephemeral-storage`, `requests.cpu`, `requests.memory` and `requests.ephemeral-storage`) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.podTemplate.env[].valueFrom.secretKeyRef`

Description

Selects a key of a secret in the pod's namespace

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

`.spec.podTemplate.hostAliases`

Description

HostAliases is an optional list of hosts and IPs that will be injected into the pod's hosts file if specified. This is only valid for non-hostNetwork pods.

Type

`array`

`.spec.podTemplate.hostAliases[]`

Description

HostAlias holds the mapping between IP and hostnames that will be injected as an entry in the pod's hosts file.

Type

`object`

Required

`ip`

Property	Type	Description
<code>hostnames</code>	<code>array</code>	Hostnames for the above IP address.

Property	Type	Description
<code>ip</code>	<code>string</code>	IP address of the host file entry.

`.spec.podTemplate.hostAliases[].hostnames`

Description

Hostnames for the above IP address.

Type

`array`

`.spec.podTemplate.hostAliases[].hostnames[]`

Type

`string`

`.spec.podTemplate.imagePullSecrets`

Description

ImagePullSecrets gives the name of the secret used by the pod to pull the image if specified

Type

`array`

`.spec.podTemplate.imagePullSecrets[]`

Description

LocalObjectReference contains enough information to let you locate the referenced object inside the same namespace.

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.podTemplate.nodeSelector

Description

NodeSelector is a selector which must be true for the pod to fit on a node. Selector which must match a node's labels for the pod to be scheduled on that node. More info: <https://kubernetes.io/docs/concepts/configuration/assign-pod-node/>

Type

object

.spec.podTemplate.tolerations

Description

If specified, the pod's tolerations.

Type

array

.spec.podTemplate.tolerations[]

Description

The pod this Toleration is attached to tolerates any taint that matches the triple <key,value,effect> using the matching operator <operator>.

Type

object

Property	Type	Description
effect	string	Effect indicates the taint effect to match. Empty means match all taint effects. When specified, allowed values are NoSchedule, PreferNoSchedule and NoExecute.
key	string	Key is the taint key that the toleration applies to. Empty means match all taint keys. If the key is empty, operator must be Exists; this combination means to match all values and all keys.
operator	string	Operator represents a key's relationship to the value. Valid operators are Exists and Equal. Defaults to Equal. Exists is equivalent to wildcard for value, so that a pod can tolerate all taints of a particular category.
tolerationSeconds	integer	TolerationSeconds represents the period of time the toleration (which must be of effect NoExecute, otherwise this field is ignored) tolerates the taint. By default, it is not set, which means tolerate the taint forever (do not evict). Zero and negative values will be treated as 0 (evict immediately) by the system.
value	string	Value is the taint value the toleration matches to. If the operator is Exists, the value should be empty, otherwise just a regular string.

.spec.podTemplate.topologySpreadConstraints

Description

TopologySpreadConstraints controls how Pods are spread across your cluster among failure-domains such as regions, zones, nodes, and other user-defined topology domains.

Type

array

.spec.podTemplate.topologySpreadConstraints[]

Description

TopologySpreadConstraint specifies how to spread matching pods among the given topology.

Type

object

Required

maxSkew

topologyKey

whenUnsatisfiable

Property	Type	Description
labelSelector	object	LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.
matchLabelKeys	array	MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is

Property	Type	Description
		forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).
maxSkew	integer	MaxSkew describes the degree to which pods may be unevenly distributed. When <code>whenUnsatisfiable=DoNotSchedule</code>, it is the maximum permitted difference between the number of matching pods in the target topology and the global minimum. The global minimum is the minimum number of matching pods in an eligible domain or zero if the number of eligible domains is less than MinDomains. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 2/2/1: In this case, the global minimum is 1. zone1 zone2 zone3 P P P P P - if MaxSkew is 1, incoming pod can only be scheduled to zone3 to become 2/2/2; scheduling it onto zone1(zone2) would make the ActualSkew(3-1) on zone1(zone2) violate MaxSkew(1). - if MaxSkew is 2, incoming pod can be scheduled onto any zone. When <code>whenUnsatisfiable=ScheduleAnyway</code>, it is used to give higher precedence to topologies

Property	Type	Description
		that satisfy it. It's a required field. Default value is 1 and 0 is not allowed.
minDomains	integer	MinDomains indicates a minimum number of eligible domains. When the number of eligible domains with matching topology keys is less than minDomains, Pod Topology Spread treats "global minimum" as 0, and then the calculation of Skew is performed. And when the number of eligible domains with matching topology keys equals or greater than minDomains, this value has no effect on scheduling. As a result, when the number of eligible domains is less than minDomains, scheduler won't schedule more than maxSkew Pods to those domains. If value is nil, the constraint behaves as if MinDomains is equal to 1. Valid values are integers greater than 0. When value is not nil, WhenUnsatisfiable must be DoNotSchedule. For example, in a 3-zone cluster, MaxSkew is set to 2, MinDomains is set to 5 and pods with the same labelSelector spread as 2/2/2: zone1 zone2 zone3 P P P P P P The number of domains is less than 5(MinDomains), so "global minimum" is treated as 0. In this situation, new pod with the same labelSelector cannot be scheduled, because computed skew will be 3(3 - 0) if new Pod is scheduled to any of the three zones, it will violate MaxSkew.
nodeAffinityPolicy	string	NodeAffinityPolicy indicates how we will treat Pod's nodeAffinity/nodeSelector when calculating pod topology spread skew. Options are:

Property	Type	Description
		• Honor: only nodes matching nodeAffinity/nodeSelector are included in the calculations. • Ignore: nodeAffinity/nodeSelector are ignored. All nodes are included in the calculations. If this value is nil, the behavior is equivalent to the Honor policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
nodeTaintsPolicy	string	NodeTaintsPolicy indicates how we will treat node taints when calculating pod topology spread skew. Options are: • Honor: nodes without taints, along with tainted nodes for which the incoming pod has a toleration, are included. • Ignore: node taints are ignored. All nodes are included. If this value is nil, the behavior is equivalent to the Ignore policy. This is a beta-level feature default enabled by the NodeInclusionPolicyInPodTopologySpread feature flag.
topologyKey	string	TopologyKey is the key of node labels. Nodes that have a label with this key and identical values are considered to be in the same topology. We consider each <key, value> as a "bucket", and try to put

Property	Type	Description
		balanced number of pods into each bucket. We define a domain as a particular instance of a topology. Also, we define an eligible domain as a domain whose nodes meet the requirements of nodeAffinityPolicy and nodeTaintsPolicy. e.g. If TopologyKey is "kubernetes.io/hostname", each Node is a domain of that topology. And, if TopologyKey is "topology.kubernetes.io/zone", each zone is a domain of that topology. It's a required field.
whenUnsatisfiable	string	WhenUnsatisfiable indicates how to deal with a pod if it doesn't satisfy the spread constraint. - DoNotSchedule (default) tells the scheduler not to schedule it. - ScheduleAnyway tells the scheduler to schedule the pod in any location, but giving higher precedence to topologies that would help reduce the skew. A constraint is considered "Unsatisfiable" for an incoming pod if and only if every possible node assignment for that pod would violate "MaxSkew" on some topology. For example, in a 3-zone cluster, MaxSkew is set to 1, and pods with the same labelSelector spread as 3/1/1: zone1 zone2 zone3 P P P P P If WhenUnsatisfiable is set to DoNotSchedule, incoming pod can only be scheduled to zone2(zone3) to become 3/2/1(3/1/2) as ActualSkew(2-1) on zone2(zone3) satisfies MaxSkew(1). In other words, the cluster can still be imbalanced, but scheduler won't make it <i>more</i> imbalanced. It's a required field.

`.spec.podTemplate.topologySpreadConstraints[].labelSelector`

Description

LabelSelector is used to find matching pods. Pods that match this label selector are counted to determine the number of pods in their corresponding topology domain.

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	object	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions`

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

key

operator

Property	Type	Description
key	string	key is the label key that the selector applies to.
operator	string	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	array	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

array

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchExpressions[].values[]`

Type

string

`.spec.podTemplate.topologySpreadConstraints[].labelSelector.matchLabels`

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

`.spec.podTemplate.topologySpreadConstraints[].matchLabelKeys`

Description

MatchLabelKeys is a set of pod label keys to select the pods over which spreading will be calculated. The keys are used to lookup values from the incoming pod labels, those key-value labels are ANDed with labelSelector to select the group of existing pods over which spreading will be calculated for the incoming pod. The same key is forbidden to exist in both MatchLabelKeys and LabelSelector. MatchLabelKeys cannot be set when LabelSelector isn't set. Keys that don't exist in the incoming pod labels will be ignored. A null or empty list means only match against labelSelector. This is a beta field and requires the MatchLabelKeysInPodTopologySpread feature gate to be enabled (enabled by default).

Type

array

.spec.podTemplate.topologySpreadConstraints[].matchLabelsKeys[]

Type

`string`

.spec.ref

Description

TaskRef can be used to refer to a specific instance of a task.

Type

`object`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task
<code>bundle</code>	<code>string</code>	Bundle url reference to a Tekton Bundle. Deprecated: Please use ResolverRef with the bundles resolver instead. The field is staying there for go client backward compatibility, but is not used/allowed anymore.
<code>kind</code>	<code>string</code>	TaskKind indicates the Kind of the Task: 1. Namespaced Task when Kind is set to "Task". If Kind is "", it defaults to "Task". 2. Custom Task when Kind is non-empty and APIVersion is non-empty

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names ↗
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

`.spec.ref.params`

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

`array`

`.spec.ref.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

name

value

Property	Type	Description
name	string	
value		

.spec.spec

Description

Spec is a specification of a custom task

Type

object

Property	Type	Description
apiVersion	string	
kind	string	
metadata	ObjectMeta	PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask
spec	object	Spec is a specification of a custom task

.spec.spec.spec

Description

Spec is a specification of a custom task

Type

object

.spec.workspaces

Description

Workspaces is a list of WorkspaceBindings from volumes to workspaces.

Type

array

.spec.workspaces[]

Description

WorkspaceBinding maps a Task's declared workspace to a Volume.

Type

object

Required

name

Property	Type	Description
<code>configMap</code>	<code>object</code>	ConfigMap represents a configMap that should populate workspace.
<code>csi</code>	<code>object</code>	CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.
<code>emptyDir</code>	<code>object</code>	EmptyDir represents a temporary directory that shares Task's lifetime. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir ↗ Either this OR PersistentVolumeClaim can be used.

Property	Type	Description
<code>name</code>	<code>string</code>	Name is the name of the workspace populated by the v
<code>persistentVolumeClaim</code>	<code>object</code>	PersistentVolumeClaimVolumeSource represents a ref to a PersistentVolumeClaim in the same namespace. E this OR EmptyDir can be used.
<code>projected</code>	<code>object</code>	Projected represents a projected volume that should pc this workspace.
<code>secret</code>	<code>object</code>	Secret represents a secret that should populate this workspace.
<code>subPath</code>	<code>string</code>	SubPath is optionally a directory on the volume which s be used for this binding (i.e. the volume will be mounter sub directory).
<code>volumeClaimTemplate</code>		VolumeClaimTemplate is a template for a claim that will created in the same namespace. The PipelineRun cont responsible for creating a unique claim for each instanc PipelineRun. See PersistentVolumeClaim (API version:

`.spec.workspaces[].configMap`

Description

ConfigMap represents a configMap that should populate this workspace.

Type

object

Property	Type	Description
defaultMode	integer	defaultMode is optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
items	array	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	optional specify whether the ConfigMap or its keys must be defined

`.spec.workspaces[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key` `path`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the key to project.

Property	Type	Description
<code>mode</code>	<code>integer</code>	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.workspaces[].csi`

Description

CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.

Type

`object`

Required

`driver`

Property	Type	Description
<code>driver</code>	<code>string</code>	driver is the name of the CSI driver that handles this volume. Consult with your admin for the correct name as registered in the cluster.

Property	Type	Description
<code>fsType</code>	<code>string</code>	fsType to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.
<code>nodePublishSecretRef</code>	<code>object</code>	nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.
<code>readOnly</code>	<code>boolean</code>	readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).
<code>volumeAttributes</code>	<code>object</code>	volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

`.spec.workspaces[].csi.nodePublishSecretRef`

Description

nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.workspaces[].csi.volumeAttributes

Description

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

Type

object

.spec.workspaces[].emptyDir

Description

EmptyDir represents a temporary directory that shares a Task's lifetime. More info: <https://kubernetes.io/docs/concepts/storage/volumes#emptydir> Either this OR PersistentVolumeClaim can be used.

Type

object

Property	Type	Description
medium	string	medium represents what type of storage medium should back this directory. The default is "" which means to use the node's

Property	Type	Description
		default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir
sizeLimit		sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

.spec.workspaces[].persistentVolumeClaim

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.

Type

object

Required

claimName

Property	Type	Description
claimName	string	claimName is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
<code>readOnly</code>	<code>boolean</code>	readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

`.spec.workspaces[].projected`

Description

Projected represents a projected volume that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	defaultMode are the mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>sources</code>	<code>array</code>	sources is the list of volume projections. Each entry in this list handles one source.

`.spec.workspaces[].projected.sources`

Description

`sources` is the list of volume projections. Each entry in this list handles one source.

Type

array

`.spec.workspaces[].projected.sources[]`

Description

Projection that may be projected along with other supported volume types. Exactly one of these fields must be set.

Type

object

Property	Type	Description
		ClusterTrustBundle allows a pod to access the <code>.spec.trustBundle</code> field of ClusterTrustBundle objects in an auto-updating file.
		Alpha, gated by the ClusterTrustBundleProjection feature gate.
<code>clusterTrustBundle</code>	object	ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector.
		Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.

Property	Type	Description
<code>configMap</code>	<code>object</code>	configMap information about the configMap data to project
<code>downwardAPI</code>	<code>object</code>	downwardAPI information about the downwardAPI data to project
<code>secret</code>	<code>object</code>	secret information about the secret data to project
<code>serviceAccountToken</code>	<code>object</code>	serviceAccountToken is information about the serviceAccountToken data to project

`.spec.workspaces[].projected.sources[].clusterTrustBundle`

Description

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.`

Type

`object`

Required

`path`

Property	Type	Description
<code>labelSelector</code>	<code>object</code>	Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".
<code>name</code>	<code>string</code>	Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.
<code>optional</code>	<code>boolean</code>	If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.
<code>path</code>	<code>string</code>	Relative path from the volume root to write the bundle.
<code>signerName</code>	<code>string</code>	Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all selected ClusterTrustBundles will be unified and deduplicated.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector`

Description

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	object	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions`

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

key

operator

Property	Type	Description
key	string	key is the label key that the selector applies to.
operator	string	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	array	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values`

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

array

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values[]`

Type

string

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchLabels`

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

`.spec.workspaces[].projected.sources[].configMap`

Description

configMap information about the configMap data to project

Type

object

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Property	Type	Description
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	optional specify whether the ConfigMap or its keys must be defined

`.spec.workspaces[].projected.sources[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].projected.sources[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.spec.workspaces[].projected.sources[].downwardAPI

Description

downwardAPI information about the downwardAPI data to project

Type

object

Property	Type	Description
items	array	Items is a list of DownwardAPIVolume file

`.spec.workspaces[].projected.sources[].downwardAPI.items`

Description

Items is a list of DownwardAPIVolume file

Type

array

`.spec.workspaces[].projected.sources[].downwardAPI.items[]`

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

object

Required

path

Property	Type	Description
<code>fieldRef</code>	<code>object</code>	Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Property	Type	Description
<code>mode</code>	<code>integer</code>	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

`.spec.workspaces[].projected.sources[].downwardAPI.items[].fieldRef`

Description

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

`.spec.workspaces[].projected.sources[].downwardAPI.items[].resourceFieldRef`

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

`object`

Required

`resource`

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"

Property	Type	Description
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.workspaces[].projected.sources[].secret`

Description

secret information about the secret data to project

Type

`object`

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its key must be defined

`.spec.workspaces[].projected.sources[].secret.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].projected.sources[].secret.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key`

`path`

Property	Type	Description
<code>key</code>	<code>string</code>	key is the key to project.

Property	Type	Description
<code>mode</code>	<code>integer</code>	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.workspaces[].projected.sources[].serviceAccountToken`

Description

serviceAccountToken is information about the serviceAccountToken data to project

Type

`object`

Required

`path`

Property	Type	Description
<code>audience</code>	<code>string</code>	audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.
<code>expirationSeconds</code>	<code>integer</code>	expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token. The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.
<code>path</code>	<code>string</code>	path is the path relative to the mount point of the file to project the token into.

`.spec.workspaces[].secret`

Description

Secret represents a secret that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> is Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code>, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	<code>items</code> If unspecified, each key-value pair in the <code>Data</code> field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the <code>..</code> path or start with <code>..</code>.
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its keys must be defined
<code>secretName</code>	<code>string</code>	<code>secretName</code> is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

.spec.workspaces[].secret.items

Description

items If unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.workspaces[].secret.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

Property	Type	Description
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.status

Description

RunStatus defines the observed state of Run

Type

`object`

Property	Type	Description
<code>annotations</code>	<code>object</code>	Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.
<code>completionTime</code>	<code>string</code>	CompletionTime is the time the build completed.
<code>conditions</code>	<code>array</code>	Conditions the latest available observations of a resource's current state.
<code>extraFields</code>	<code>object</code>	ExtraFields holds arbitrary fields provided by the custom task controller.

Property	Type	Description
<code>observedGeneration</code>	<code>integer</code>	ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.
<code>results</code>	<code>array</code>	Results reports any output result values to be consumed by later tasks in a pipeline.
<code>retriesStatus</code>	<code>array</code>	RetriesStatus contains the history of RunStatus, in case of a retry.
<code>startTime</code>	<code>string</code>	StartTime is the time the build is actually started.

.status.annotations

Description

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

Type

`object`

.status.conditions

Description

Conditions the latest available observations of a resource's current state.

Type

`array`

.status.conditions[]

Description

Condition defines a readiness condition for a Knative resource. See: <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties>

Type

`object`

Required

`status``type`

Property	Type	Description
<code>lastTransitionTime</code>	<code>string</code>	LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).
<code>message</code>	<code>string</code>	A human readable message indicating details about the transition.
<code>reason</code>	<code>string</code>	The reason for the condition's last transition.
<code>severity</code>	<code>string</code>	Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

Property	Type	Description
<code>status</code>	<code>string</code>	Status of the condition, one of True, False, Unknown.
<code>type</code>	<code>string</code>	Type of condition.

`.status.extraFields`

Description

ExtraFields holds arbitrary fields provided by the custom task controller.

Type

`object`

`.status.results`

Description

Results reports any output result values to be consumed by later tasks in a pipeline.

Type

`array`

`.status.results[]`

Description

RunResult used to describe the results of a task

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	Name the given name
<code>value</code>	<code>string</code>	Value the given value of the result

`.status.retriesStatus`

Description

RetriesStatus contains the history of RunStatus, in case of a retry.

Type

`array`

`.status.retriesStatus[]`

Type

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/runs`
 - `DELETE` : delete collection of Run
 - `GET` : list objects of kind Run
 - `POST` : create a new Run
- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/runs/{name}`
 - `DELETE` : delete the specified Run
 - `GET` : read the specified Run

- **PATCH** : partially update the specified Run
- **PUT** : replace the specified Run
- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/runs/{name}/status`
 - **GET** : read status of the specified Run
 - **PATCH** : partially update status of the specified Run
 - **PUT** : replace status of the specified Run

`/apis/tekton.dev/v1alpha1/namespaces/{namespace}/runs`

HTTP method

DELETE

Description

delete collection of Run

HTTP responses

HTTP code	Response body
200 - OK	Status  schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind Run

HTTP responses

HTTP code	Response body
200 - OK	RunList  schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new Run

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Run</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	Run schema
201 - Created	Run schema
202 - Accepted	Run schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1alpha1/namespaces/{namespace}/runs/{name}

HTTP method

[DELETE](#)

Description

delete the specified Run

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	Status  schema
202 - Accepted	Status  schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified Run

HTTP responses

HTTP code	Response body
200 - OK	<code>Run</code> schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified Run

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a

Parameter	Type	Description
		BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>Run</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified Run

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only

Parameter	Type	Description
		persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Run</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>Run</code> schema
201 - Created	<code>Run</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1alpha1/namespaces/{namespace}/runs/{name}/status

HTTP method

`GET`

Description

read status of the specified Run

HTTP responses

HTTP code	Response body
200 - OK	<code>Run</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified Run

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>Run</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified Run

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>Run</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>Run</code> schema
201 - Created	<code>Run</code> schema
401 - Unauthorized	Empty

CustomRun [tekton.dev/v1]

Description

CustomRun represents a single execution of a Custom Task.

Type

object

Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗

Property	Type	Description
<code>metadata</code>	<code>ObjectMeta</code> ↗	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	CustomRunSpec defines the desired state of CustomRun
<code>status</code>	<code>object</code>	CustomRunStatus defines the observed state of CustomRun

.spec

Description

CustomRunSpec defines the desired state of CustomRun

Type

`object`

Property	Type	Description
<code>customRef</code>	<code>object</code>	TaskRef can be used to refer to a specific instance of a task.
<code>customSpec</code>	<code>object</code>	Spec is a specification of a custom task
<code>params</code>	<code>array</code>	Params is a list of Param

Property	Type	Description
<code>retries</code>	<code>integer</code>	Used for propagating retries count to custom tasks
<code>serviceAccountName</code>	<code>string</code>	
<code>status</code>	<code>string</code>	Used for cancelling a customrun (and maybe more later on)
<code>statusMessage</code>	<code>string</code>	Status message for cancellation.
<code>timeout</code>	<code>string</code>	Time after which the custom-task times out. Refer Go's ParseDuration documentation for expected format: https://golang.org/pkg/time/#ParseDuration ↗
<code>workspaces</code>	<code>array</code>	Workspaces is a list of WorkspaceBindings from volumes to workspaces.

.spec.customRef

Description

TaskRef can be used to refer to a specific instance of a task.

Type

`object`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task
<code>bundle</code>	<code>string</code>	Bundle url reference to a Tekton Bundle. Deprecated: Please use ResolverRef with the bundles resolver instead. The field is staying there for go client backward compatibility, but is not used/allowed anymore.
<code>kind</code>	<code>string</code>	TaskKind indicates the Kind of the Task: 1. Namespaced Task when Kind is set to "Task". If Kind is "", it defaults to "Task". 2. Custom Task when Kind is non-empty and APIVersion is non-empty
<code>name</code>	<code>string</code>	Name of the referent; More info: http://kubernetes.io/docs/user-guide/identifiers#names
<code>params</code>	<code>array</code>	Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.
<code>resolver</code>	<code>string</code>	Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

.spec.customRef.params

Description

Params contains the parameters used to identify the referenced Tekton resource. Example entries might include "repo" or "path" but the set of params ultimately depends on the chosen resolver.

Type

array

.spec.customRef.params[]

Description

Param declares an ParamValues to use for the parameter called name.

Type

object

Required

name

value

Property	Type	Description
name	string	
value		

.spec.customSpec

Description

Spec is a specification of a custom task

Type

object

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	
<code>kind</code>	<code>string</code>	
<code>metadata</code>	ObjectMeta	PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask
<code>spec</code>	<code>object</code>	Spec is a specification of a custom task

`.spec.customSpec.spec`

Description

Spec is a specification of a custom task

Type

`object`

`.spec.params`

Description

Params is a list of Param

Type

`array`

`.spec.params[]`

Description

Param declares an ParamValues to use for the parameter called name.

Type

`object`

Required

`name``value`

Property	Type	Description
<code>name</code>	<code>string</code>	
<code>value</code>		

`.spec.workspaces`

Description

Workspaces is a list of WorkspaceBindings from volumes to workspaces.

Type

`array`

`.spec.workspaces[]`

Description

WorkspaceBinding maps a Task's declared workspace to a Volume.

Type

`object`

Required

`name`

Property	Type	Description
<code>configMap</code>	<code>object</code>	ConfigMap represents a configMap that should populate workspace.

Property	Type	Description
<code>csi</code>	<code>object</code>	CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.
<code>emptyDir</code>	<code>object</code>	EmptyDir represents a temporary directory that shares Task's lifetime. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir Either this OR PersistentVolumeClaim can be used.
<code>name</code>	<code>string</code>	Name is the name of the workspace populated by the volume.
<code>persistentVolumeClaim</code>	<code>object</code>	PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.
<code>projected</code>	<code>object</code>	Projected represents a projected volume that should populate this workspace.
<code>secret</code>	<code>object</code>	Secret represents a secret that should populate this workspace.
<code>subPath</code>	<code>string</code>	SubPath is optionally a directory on the volume which should be used for this binding (i.e. the volume will be mounted to the sub directory).

Property	Type	Description
<code>volumeClaimTemplate</code>	<code>object</code>	VolumeClaimTemplate is a template for a claim that will be created in the same namespace. The PipelineRun controller is responsible for creating a unique claim for each instance of PipelineRun. See PersistentVolumeClaim (API version: v1).

`.spec.workspaces[].configMap`

Description

ConfigMap represents a configMap that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	defaultMode is optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume

Property	Type	Description
		setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
optional	boolean	optional specify whether the ConfigMap or its keys must be defined

.spec.workspaces[].configMap.items

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.workspaces[].configMap.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.spec.workspaces[].csi

Description

CSI (Container Storage Interface) represents ephemeral storage that is handled by certain external CSI drivers.

Type

object

Required

driver

Property	Type	Description
<code>driver</code>	<code>string</code>	driver is the name of the CSI driver that handles this volume. Consult with your admin for the correct name as registered in the cluster.
<code>fsType</code>	<code>string</code>	fsType to mount. Ex. "ext4", "xfs", "ntfs". If not provided, the empty value is passed to the associated CSI driver which will determine the default filesystem to apply.
<code>nodePublishSecretRef</code>	<code>object</code>	nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.
<code>readOnly</code>	<code>boolean</code>	readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).
<code>volumeAttributes</code>	<code>object</code>	volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

.spec.workspaces[].csi.nodePublishSecretRef

Description

nodePublishSecretRef is a reference to the secret object containing sensitive information to pass to the CSI driver to complete the CSI NodePublishVolume and NodeUnpublishVolume calls. This field is optional, and may be empty if no secret is required. If the secret object contains more than one secret, all secret references are passed.

Type

object

Property	Type	Description
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

.spec.workspaces[].csi.volumeAttributes

Description

volumeAttributes stores driver-specific properties that are passed to the CSI driver. Consult your driver's documentation for supported values.

Type

object

.spec.workspaces[].emptyDir

Description

EmptyDir represents a temporary directory that shares a Task's lifetime. More info: <https://kubernetes.io/docs/concepts/storage/volumes#emptydir> Either this OR PersistentVolumeClaim can be used.

Type

object

Property	Type	Description
<code>medium</code>	<code>string</code>	medium represents what type of storage medium should back this directory. The default is "" which means to use the node's default medium. Must be an empty string (default) or Memory. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir
<code>sizeLimit</code>		sizeLimit is the total amount of local storage required for this EmptyDir volume. The size limit is also applicable for memory medium. The maximum usage on memory medium EmptyDir would be the minimum value between the SizeLimit specified here and the sum of memory limits of all containers in a pod. The default is nil which means that the limit is undefined. More info: https://kubernetes.io/docs/concepts/storage/volumes#emptydir

`.spec.workspaces[].persistentVolumeClaim`

Description

PersistentVolumeClaimVolumeSource represents a reference to a PersistentVolumeClaim in the same namespace. Either this OR EmptyDir can be used.

Type

`object`

Required

`claimName`

Property	Type	Description
<code>claimName</code>	<code>string</code>	<code>claimName</code> is the name of a PersistentVolumeClaim in the same namespace as the pod using this volume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#persistentvolumeclaims ↗
<code>readOnly</code>	<code>boolean</code>	<code>readOnly</code> Will force the ReadOnly setting in VolumeMounts. Default false.

`.spec.workspaces[].projected`

Description

Projected represents a projected volume that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> are the mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code> , and the result can be other mode bits set.

Property	Type	Description
<code>sources</code>	<code>array</code>	<code>sources</code> is the list of volume projections. Each entry in this list handles one source.

`.spec.workspaces[].projected.sources`

Description

`sources` is the list of volume projections. Each entry in this list handles one source.

Type

`array`

`.spec.workspaces[].projected.sources[]`

Description

Projection that may be projected along with other supported volume types. Exactly one of these fields must be set.

Type

`object`

Property	Type	Description
<code>clusterTrustBundle</code>	<code>object</code>	ClusterTrustBundle allows a pod to access the <code>.spec.trustBundle</code> field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector.

Property	Type	Description
		Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem. Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.
<code>configMap</code>	<code>object</code>	configMap information about the configMap data to project
<code>downwardAPI</code>	<code>object</code>	downwardAPI information about the downwardAPI data to project
<code>secret</code>	<code>object</code>	secret information about the secret data to project
<code>serviceAccountToken</code>	<code>object</code>	serviceAccountToken is information about the serviceAccountToken data to project

`.spec.workspaces[].projected.sources[].clusterTrustBundle`

Description

ClusterTrustBundle allows a pod to access the `.spec.trustBundle` field of ClusterTrustBundle objects in an auto-updating file. Alpha, gated by the ClusterTrustBundleProjection feature gate. ClusterTrustBundle objects can either be selected by name, or by the combination of signer name and a label selector. Kubelet performs aggressive normalization of the PEM contents written into the pod filesystem.`

Esoteric PEM features such as inter-block comments and block headers are stripped. Certificates are deduplicated. The ordering of certificates within the file is arbitrary, and Kubelet may change the order over time.

Type

object

Required

path

Property	Type	Description
labelSelector	object	Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".
name	string	Select a single ClusterTrustBundle by object name. Mutually-exclusive with signerName and labelSelector.
optional	boolean	If true, don't block pod startup if the referenced ClusterTrustBundle(s) aren't available. If using name, then the named ClusterTrustBundle is allowed not to exist. If using signerName, then the combination of signerName and labelSelector is allowed to match zero ClusterTrustBundles.
path	string	Relative path from the volume root to write the bundle.
signerName	string	Select all ClusterTrustBundles that match this signer name. Mutually-exclusive with name. The contents of all

Property	Type	Description
		selected ClusterTrustBundles will be unified and deduplicated.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector`

Description

Select all ClusterTrustBundles that match this label selector. Only has effect if signerName is set. Mutually-exclusive with name. If unset, interpreted as "match nothing". If set but empty, interpreted as "match everything".

Type

object

Property	Type	Description
<code>matchExpressions</code>	array	matchExpressions is a list of label selector requirements. The requirements are ANDed.
<code>matchLabels</code>	object	matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions`

Description

matchExpressions is a list of label selector requirements. The requirements are ANDed.

Type

array

`.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[]`

Description

A label selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

Type

object

Required

key

operator

Property	Type	Description
key	string	key is the label key that the selector applies to.
operator	string	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.
values	array	values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values

Description

values is an array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

Type

array

.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchExpressions[].values[]

Type

string

.spec.workspaces[].projected.sources[].clusterTrustBundle.labelSelector.matchLabels

Description

matchLabels is a map of {key,value} pairs. A single {key,value} in the matchLabels map is equivalent to an element of matchExpressions, whose key field is "key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

Type

object

.spec.workspaces[].projected.sources[].configMap

Description

configMap information about the configMap data to project

Type

object

Property	Type	Description
items	array	items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
name	string	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
optional	boolean	optional specify whether the ConfigMap or its keys must be defined

`.spec.workspaces[].projected.sources[].configMap.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced ConfigMap will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the ConfigMap, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

`.spec.workspaces[].projected.sources[].configMap.items[]`

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

`.spec.workspaces[].projected.sources[].downwardAPI`

Description

downwardAPI information about the downwardAPI data to project

Type

object

Property	Type	Description
items	array	Items is a list of DownwardAPIVolume file

`.spec.workspaces[].projected.sources[].downwardAPI.items`

Description

Items is a list of DownwardAPIVolume file

Type

array

`.spec.workspaces[].projected.sources[].downwardAPI.items[]`

Description

DownwardAPIVolumeFile represents information to create the file containing the pod field

Type

object

Required

path

Property	Type	Description
<code>fieldRef</code>	<code>object</code>	Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.
<code>mode</code>	<code>integer</code>	Optional: mode bits used to set permissions on this file, must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
<code>path</code>	<code>string</code>	Required: Path is the relative path name of the file to be created. Must not be absolute or contain the '..' path. Must be utf-8 encoded. The first item of the relative path must not start with '..'
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

`.spec.workspaces[].projected.sources[].downwardAPI.items[].fieldRef`

Description

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

Type

object

Required

fieldPath

Property	Type	Description
apiVersion	string	Version of the schema the FieldPath is written in terms of, defaults to "v1".
fieldPath	string	Path of the field to select in the specified API version.

.spec.workspaces[].projected.sources[].downwardAPI.items[].resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

Type

object

Required

resource

Property	Type	Description
containerName	string	Container name: required for volumes, optional for env vars

Property	Type	Description
<code>divisor</code>	<code>integer</code>	Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

`.spec.workspaces[].projected.sources[].secret`

Description

secret information about the secret data to project

Type

`object`

Property	Type	Description
<code>items</code>	<code>array</code>	items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its key must be defined

`.spec.workspaces[].projected.sources[].secret.items`

Description

items if unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

`array`

`.spec.workspaces[].projected.sources[].secret.items[]`

Description

Maps a string key to a path within a volume.

Type

`object`

Required

`key`

`path`

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.
path	string	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.spec.workspaces[].projected.sources[].serviceAccountToken

Description

serviceAccountToken is information about the serviceAccountToken data to project

Type

object

Required

path

Property	Type	Description
<code>audience</code>	<code>string</code>	audience is the intended audience of the token. A recipient of a token must identify itself with an identifier specified in the audience of the token, and otherwise should reject the token. The audience defaults to the identifier of the apiserver.
<code>expirationSeconds</code>	<code>integer</code>	expirationSeconds is the requested duration of validity of the service account token. As the token approaches expiration, the kubelet volume plugin will proactively rotate the service account token. The kubelet will start trying to rotate the token if the token is older than 80 percent of its time to live or if the token is older than 24 hours.Defaults to 1 hour and must be at least 10 minutes.
<code>path</code>	<code>string</code>	path is the path relative to the mount point of the file to project the token into.

`.spec.workspaces[].secret`

Description

Secret represents a secret that should populate this workspace.

Type

`object`

Property	Type	Description
<code>defaultMode</code>	<code>integer</code>	<code>defaultMode</code> is Optional: mode bits used to set permissions on created files by default. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. Defaults to 0644. Directories within the path are not affected by this setting. This might be in conflict with other options that affect the file mode, like <code>fsGroup</code>, and the result can be other mode bits set.
<code>items</code>	<code>array</code>	<code>items</code> If unspecified, each key-value pair in the <code>Data</code> field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the <code>..</code> path or start with <code>..</code>.
<code>optional</code>	<code>boolean</code>	optional field specify whether the Secret or its keys must be defined
<code>secretName</code>	<code>string</code>	<code>secretName</code> is the name of the secret in the pod's namespace to use. More info: https://kubernetes.io/docs/concepts/storage/volumes#secret

.spec.workspaces[].secret.items

Description

items If unspecified, each key-value pair in the Data field of the referenced Secret will be projected into the volume as a file whose name is the key and content is the value. If specified, the listed keys will be projected into the specified paths, and unlisted keys will not be present. If a key is specified which is not present in the Secret, the volume setup will error unless it is marked optional. Paths must be relative and may not contain the '..' path or start with '..'.

Type

array

.spec.workspaces[].secret.items[]

Description

Maps a string key to a path within a volume.

Type

object

Required

key

path

Property	Type	Description
key	string	key is the key to project.
mode	integer	mode is Optional: mode bits used to set permissions on this file. Must be an octal value between 0000 and 0777 or a decimal value between 0 and 511. YAML accepts both octal and decimal values, JSON requires decimal values for mode bits. If not specified, the volume defaultMode will be used. This might be in conflict with other options that affect the file mode, like fsGroup, and the result can be other mode bits set.

Property	Type	Description
<code>path</code>	<code>string</code>	path is the relative path of the file to map the key to. May not be an absolute path. May not contain the path element '..'. May not start with the string '..'.

.status

Description

CustomRunStatus defines the observed state of CustomRun

Type

`object`

Property	Type	Description
<code>annotations</code>	<code>object</code>	Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.
<code>completionTime</code>	<code>string</code>	CompletionTime is the time the build completed.
<code>conditions</code>	<code>array</code>	Conditions the latest available observations of a resource's current state.
<code>extraFields</code>		ExtraFields holds arbitrary fields provided by the custom task controller.

Property	Type	Description
<code>observedGeneration</code>	<code>integer</code>	ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.
<code>results</code>	<code>array</code>	Results reports any output result values to be consumed by later tasks in a pipeline.
<code>retriesStatus</code>	<code></code>	RetriesStatus contains the history of CustomRunStatus, in case of a retry. See CustomRun.status (API version: tekton.dev/v1beta1)
<code>startTime</code>	<code>string</code>	StartTime is the time the build is actually started.

.status.annotations

Description

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

Type

`object`

.status.conditions

Description

Conditions the latest available observations of a resource's current state.

Type

array

.status.conditions[]

Description

Condition defines a readiness condition for a Knative resource. See: <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties>

Type

object

Required

status

type

Property	Type	Description
lastTransitionTime	string	LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).
message	string	A human readable message indicating details about the transition.
reason	string	The reason for the condition's last transition.
severity	string	Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

Property	Type	Description
<code>status</code>	<code>string</code>	Status of the condition, one of True, False, Unknown.
<code>type</code>	<code>string</code>	Type of condition.

`.status.results`

Description

Results reports any output result values to be consumed by later tasks in a pipeline.

Type

`array`

`.status.results[]`

Description

CustomRunResult used to describe the results of a task

Type

`object`

Required

`name`

`value`

Property	Type	Description
<code>name</code>	<code>string</code>	Name the given name
<code>value</code>	<code>string</code>	Value the given value of the result

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1beta1/namespaces/{namespace}/customruns`
 - `DELETE` : delete collection of CustomRun
 - `GET` : list objects of kind CustomRun
 - `POST` : create a new CustomRun
- `/apis/tekton.dev/v1beta1/namespaces/{namespace}/customruns/{name}`
 - `DELETE` : delete the specified CustomRun
 - `GET` : read the specified CustomRun
 - `PATCH` : partially update the specified CustomRun
 - `PUT` : replace the specified CustomRun
- `/apis/tekton.dev/v1beta1/namespaces/{namespace}/customruns/{name}/status`
 - `GET` : read status of the specified CustomRun
 - `PATCH` : partially update status of the specified CustomRun
 - `PUT` : replace status of the specified CustomRun

`/apis/tekton.dev/v1beta1/namespaces/{namespace}/customruns`

HTTP method

`DELETE`

Description

delete collection of CustomRun

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind CustomRun

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRunList</code> schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new CustomRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last

Parameter	Type	Description
		duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>CustomRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code> schema
201 - Created	<code>CustomRun</code> schema
202 - Accepted	<code>CustomRun</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1beta1/namespaces/{namespace}/customruns/{name}

HTTP method

DELETE

Description

delete the specified CustomRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code>  schema
202 - Accepted	<code>Status</code>  schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified CustomRun

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code>  schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified CustomRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified CustomRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>CustomRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code> schema

HTTP code	Response body
201 - Created	<code>CustomRun</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1beta1/namespaces/{namespace}/customruns/{name}/status

HTTP method

`GET`

Description

read status of the specified CustomRun

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified CustomRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+. - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified CustomRun

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing

Parameter	Type	Description
		of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>CustomRun</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>CustomRun</code> schema
201 - Created	<code>CustomRun</code> schema
401 - Unauthorized	Empty



StepAction [tekton.dev/v1]

Description

StepAction represents the actionable components of Step. The Step can only reference it from the cluster or using remote resolution.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources ↗

Property	Type	Description
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds
<code>metadata</code>	<code>ObjectMeta</code>	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the desired state of the Step from the client

.spec

Description

Spec holds the desired state of the Step from the client

Type

`object`

Property	Type	Description
<code>args</code>	<code>array</code>	Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string

Property	Type	Description
		literal "\${VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell ↗
command	array	Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references \$(VAR_NAME) are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double \$\$ are reduced to a single \$, which allows for escaping the \$(VAR_NAME) syntax: i.e. "\$\$(VAR_NAME)" will produce the string literal "\${VAR_NAME)". Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell ↗
description	string	Description is a user-facing description of the stepaction that may be used to populate a UI.
env	array	List of environment variables to set in the container. Cannot be updated.
image	string	Image reference name to run for this StepAction. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/containers/images/ ↗
params	array	Params is a list of input parameters required to run the stepAction. Params must be supplied as inputs in Steps unless they declare a defaultvalue.
results	array	Results are values that this StepAction can output
script	string	Script is the contents of an executable file to execute. If Script is not empty, the Step cannot have an Command and the Args will be passed to the Script.
securityContext	object	SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info: https://kubernetes.io/docs/tasks/configure-pod-container/security-context/ ↗ The value set in StepAction will take precedence over the value from Task.
volumeMounts	array	Volumes to mount into the Step's filesystem. Cannot be updated.

Property	Type	Description
<code>workingDir</code>	<code>string</code>	Step's working directory. If not specified, the container runtime's default will be used, which might be configured in the container image. Cannot be updated.

.spec.args

Description

Arguments to the entrypoint. The image's CMD is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e.

`$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`. Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated.

More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

`array`

.spec.args[]

Type

`string`

.spec.command

Description

Entrypoint array. Not executed within a shell. The image's ENTRYPOINT is used if this is not provided. Variable references `$(VAR_NAME)` are expanded using the container's environment. If a variable cannot be resolved, the reference in the input string will be unchanged. Double `$$` are reduced to a single `$`, which allows for escaping the `$(VAR_NAME)` syntax: i.e. `$$$(VAR_NAME)` will produce the string literal `$(VAR_NAME)`.

Escaped references will never be expanded, regardless of whether the variable exists or not. Cannot be updated. More info: <https://kubernetes.io/docs/tasks/inject-data-application/define-command-argument-container/#running-a-command-in-a-shell>

Type

array

.spec.command[]

Type

string

.spec.env

Description

List of environment variables to set in the container. Cannot be updated.

Type

array

.spec.env[]

Description

EnvVar represents an environment variable present in a Container.

Type

object

Required

name

Property	Type	Description
name	string	Name of the environment variable. Must be a C_IDENTIFIER.

Property	Type	Description
<code>value</code>	<code>string</code>	Variable references <code>\$(VAR_NAME)</code> are expanded using the previously defined environment variables in the container and any service environment variables. If a variable cannot be resolved, the reference in the input string will be unchanged. Double <code>\$\$</code> are reduced to a single <code>\$</code> , which allows for escaping the <code>\$(VAR_NAME)</code> syntax: i.e. <code>\$\$\$(VAR_NAME)</code> will produce the string literal <code>\$(VAR_NAME)</code> . Escaped references will never be expanded, regardless of whether the variable exists or not. Defaults to <code>""</code> .
<code>valueFrom</code>	<code>object</code>	Source for the environment variable's value. Cannot be used if value is not empty.

`.spec.env[].valueFrom`

Description

Source for the environment variable's value. Cannot be used if value is not empty.

Type

`object`

Property	Type	Description
<code>configMapKeyRef</code>	<code>object</code>	Selects a key of a ConfigMap.
<code>fieldRef</code>	<code>object</code>	Selects a field of the pod: supports <code>metadata.name</code> , <code>metadata.namespace</code> , <code>metadata.labels['<KEY>']</code> , <code>metadata.annotations['<KEY>']</code> , <code>spec.nodeName</code> ,

Property	Type	Description
		spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.
<code>resourceFieldRef</code>	<code>object</code>	Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.
<code>secretKeyRef</code>	<code>object</code>	Selects a key of a secret in the pod's namespace

`.spec.env[].valueFrom.configMapKeyRef`

Description

Selects a key of a ConfigMap.

Type

`object`

Required

`key`

Property	Type	Description
<code>key</code>	<code>string</code>	The key to select.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the ConfigMap or its key must be defined

.spec.env[].valueFrom.fieldRef

Description

Selects a field of the pod: supports metadata.name, metadata.namespace, `metadata.labels['<KEY>']`, `metadata.annotations['<KEY>']`, spec.nodeName, spec.serviceAccountName, status.hostIP, status.podIP, status.podIPs.

Type

`object`

Required

`fieldPath`

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	Version of the schema the FieldPath is written in terms of, defaults to "v1".
<code>fieldPath</code>	<code>string</code>	Path of the field to select in the specified API version.

.spec.env[].valueFrom.resourceFieldRef

Description

Selects a resource of the container: only resources limits and requests (limits.cpu, limits.memory, limits.ephemeral-storage, requests.cpu, requests.memory and requests.ephemeral-storage) are currently supported.

Type

object

Required

resource

Property	Type	Description
<code>containerName</code>	<code>string</code>	Container name: required for volumes, optional for env vars
<code>divisor</code>		Specifies the output format of the exposed resources, defaults to "1"
<code>resource</code>	<code>string</code>	Required: resource to select

.spec.env[].valueFrom.secretKeyRef

Description

Selects a key of a secret in the pod's namespace

Type

object

Required

key

Property	Type	Description
<code>key</code>	<code>string</code>	The key of the secret to select from. Must be a valid secret key.
<code>name</code>	<code>string</code>	Name of the referent. This field is effectively required, but due to backwards compatibility is allowed to be empty. Instances of this type with an empty value here are almost certainly wrong. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗
<code>optional</code>	<code>boolean</code>	Specify whether the Secret or its key must be defined

.spec.params

Description

Params is a list of input parameters required to run the stepAction. Params must be supplied as inputs in Steps unless they declare a defaultvalue.

Type

`array`

.spec.params[]

Description

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources). Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

Type

`object`

Required

`name`

Property	Type	Description
<code>default</code>	<code></code>	Default is the value a parameter takes if no input value is supplied. If default is set, a Task may be executed without a supplied value for the parameter.
<code>description</code>	<code>string</code>	Description is a user-facing description of the parameter that may be used to populate a UI.
<code>enum</code>	<code>array</code>	Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.
<code>name</code>	<code>string</code>	Name declares the name by which a parameter is referenced.
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs parameter.
<code>type</code>	<code>string</code>	Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

`.spec.params[].enum`

Description

Enum declares a set of allowed param input values for tasks/pipelines that can be validated. If Enum is not set, no input validation is performed for the param.

Type

array

.spec.params[].enum[]

Type

string

.spec.params[].properties

Description

Properties is the JSON Schema properties to support key-value pairs parameter.

Type

object

.spec.results

Description

Results are values that this StepAction can output

Type

array

.spec.results[]

Description

StepResult used to describe the Results of a Step.

Type

object

Required

`name`

Property	Type	Description
<code>description</code>	<code>string</code>	Description is a human-readable description of the result
<code>name</code>	<code>string</code>	Name the given name
<code>properties</code>	<code>object</code>	Properties is the JSON Schema properties to support key-value pairs results.
<code>type</code>	<code>string</code>	The possible types are 'string', 'array', and 'object', with 'string' as the default.

`.spec.results[].properties`

Description

Properties is the JSON Schema properties to support key-value pairs results.

Type

`object`

`.spec.securityContext`

Description

SecurityContext defines the security options the Step should be run with. If set, the fields of SecurityContext override the equivalent fields of PodSecurityContext. More info:

<https://kubernetes.io/docs/tasks/configure-pod-container/security-context/> The value set in StepAction will take precedence over the value from Task.

Type

object

Property	Type	Description
<code>allowPrivilegeEscalation</code>	<code>boolean</code>	AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the <code>no_new_privs</code> flag will be set on the container process. AllowPrivilegeEscalation is true always when the container is: 1. run as Privileged 2. has CAP_SYS_ADMIN Note that this field cannot be set when spec.os.name is windows.
<code>appArmorProfile</code>	<code>object</code>	appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.
<code>capabilities</code>	<code>object</code>	The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when spec.os.name is windows.
<code>privileged</code>	<code>boolean</code>	Run container in privileged mode. Processes in privileged containers are essentially equivalent to root on the host. Defaults to

Property	Type	Description
		false. Note that this field cannot be set when spec.os.name is windows.
procMount	string	procMount denotes the type of proc mount to use for the containers. The default value is Default which uses the container runtime defaults for readonly paths and masked paths. This requires the ProcMountType feature flag to be enabled. Note that this field cannot be set when spec.os.name is windows.
readOnlyRootFilesystem	boolean	Whether this container has a read-only root filesystem. Default is false. Note that this field cannot be set when spec.os.name is windows.
runAsGroup	integer	The GID to run the entrypoint of the container process. Uses runtime default if unset. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
runAsNonRoot	boolean	Indicates that the container must run as a non-root user. If true, the Kubelet will validate the image at runtime to ensure that it does not run as UID 0 (root) and fail to

Property	Type	Description
		start the container if it does. If unset or false, no such validation will be performed. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.
runAsUser	integer	The UID to run the entrypoint of the container process. Defaults to user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
seLinuxOptions	object	The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.
seccompProfile	object	The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that

Property	Type	Description
		this field cannot be set when spec.os.name is windows.
<code>windowsOptions</code>	<code>object</code>	The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

`.spec.securityContext.appArmorProfile`

Description

appArmorProfile is the AppArmor options to use by this container. If set, this profile overrides the pod's appArmorProfile. Note that this field cannot be set when spec.os.name is windows.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	localhostProfile indicates a profile loaded on the node that should be used. The profile must be preconfigured on the node to work. Must match the loaded name of the profile. Must be set if and only if type is "Localhost".

Property	Type	Description
<code>type</code>	<code>string</code>	type indicates which kind of AppArmor profile will be applied. Valid options are: Localhost - a profile pre-loaded on the node. RuntimeDefault - the container runtime's default profile. Unconfined - no AppArmor enforcement.

`.spec.securityContext.capabilities`

Description

The capabilities to add/drop when running containers. Defaults to the default set of capabilities granted by the container runtime. Note that this field cannot be set when `spec.os.name` is windows.

Type

`object`

Property	Type	Description
<code>add</code>	<code>array</code>	Added capabilities
<code>drop</code>	<code>array</code>	Removed capabilities

`.spec.securityContext.capabilities.add`

Description

Added capabilities

Type

`array`

.spec.securityContext.capabilities.add[]

Description

Capability represent POSIX capabilities type

Type

string

.spec.securityContext.capabilities.drop

Description

Removed capabilities

Type

array

.spec.securityContext.capabilities.drop[]

Description

Capability represent POSIX capabilities type

Type

string

.spec.securityContext.seLinuxOptions

Description

The SELinux context to be applied to the container. If unspecified, the container runtime will allocate a random SELinux context for each container. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is windows.

Type

object

Property	Type	Description
<code>level</code>	<code>string</code>	Level is SELinux level label that applies to the container.
<code>role</code>	<code>string</code>	Role is a SELinux role label that applies to the container.
<code>type</code>	<code>string</code>	Type is a SELinux type label that applies to the container.
<code>user</code>	<code>string</code>	User is a SELinux user label that applies to the container.

`.spec.securityContext.seccompProfile`

Description

The seccomp options to use by this container. If seccomp options are provided at both the pod & container level, the container options override the pod options. Note that this field cannot be set when `spec.os.name` is `windows`.

Type

`object`

Required

`type`

Property	Type	Description
<code>localhostProfile</code>	<code>string</code>	<code>localhostProfile</code> indicates a profile defined in a file on the node should be used. The profile must be preconfigured on the node to work. Must be a descending path, relative to the kubelet's configured

Property	Type	Description
		seccomp profile location. Must be set if type is "Localhost". Must NOT be set for any other type.
		type indicates which kind of seccomp profile will be applied. Valid options are:
type	string	Localhost - a profile defined in a file on the node should be used. RuntimeDefault - the container runtime default profile should be used. Unconfined - no profile should be applied.

.spec.securityContext.windowsOptions

Description

The Windows specific settings applied to all containers. If unspecified, the options from the PodSecurityContext will be used. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence. Note that this field cannot be set when spec.os.name is linux.

Type

object

Property	Type	Description
gmsaCredentialSpec	string	GMSACredentialSpec is where the GMSA admission webhook (https://github.com/kubernetes-sigs/windows-gmsa ↗) inlines the contents of the GMSA credential spec named by the <code>GMSACredentialSpecName</code> field.

Property	Type	Description
<code>gmsaCredentialSpecName</code>	<code>string</code>	GMSACredentialSpecName is the name of the GMSA credential spec to use.
<code>hostProcess</code>	<code>boolean</code>	HostProcess determines if a container should be run as a 'Host Process' container. All of a Pod's containers must have the same effective HostProcess value (it is not allowed to have a mix of HostProcess containers and non-HostProcess containers). In addition, if HostProcess is true then HostNetwork must also be set to true.
<code>runAsUserName</code>	<code>string</code>	The UserName in Windows to run the entrypoint of the container process. Defaults to the user specified in image metadata if unspecified. May also be set in PodSecurityContext. If set in both SecurityContext and PodSecurityContext, the value specified in SecurityContext takes precedence.

`.spec.volumeMounts`

Description

Volumes to mount into the Step's filesystem. Cannot be updated.

Type

`array`

.spec.volumeMounts[]

Description

VolumeMount describes a mounting of a Volume within a container.

Type

object

Required

mountPath

name

Property	Type	Description
<code>mountPath</code>	<code>string</code>	Path within the container at which the volume should be mounted. Must not contain ':'. <code>mountPropagation</code> determines how mounts are propagated from the host to container and the other way around. When not set, <code>MountPropagationNone</code> is used. This field is beta in 1.10. When <code>RecursiveReadOnly</code> is set to <code>IfPossible</code> or to <code>Enabled</code> , <code>MountPropagation</code> must be <code>None</code> or unspecified (which defaults to <code>None</code>).
<code>name</code>	<code>string</code>	This must match the Name of a Volume.
<code>readOnly</code>	<code>boolean</code>	Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.
<code>recursiveReadOnly</code>	<code>string</code>	<code>RecursiveReadOnly</code> specifies whether read-only mounts should be handled recursively.

Property	Type	Description
		If ReadOnly is false, this field has no meaning and must be unspecified. If ReadOnly is true, and this field is set to Disabled, the mount is not made recursively read-only. If this field is set to IfPossible, the mount is made recursively read-only, if it is supported by the container runtime. If this field is set to Enabled, the mount is made recursively read-only if it is supported by the container runtime, otherwise the pod will not be started and an error will be generated to indicate the reason. If this field is set to IfPossible or Enabled, MountPropagation must be set to None (or be unspecified, which defaults to None). If this field is not specified, it is treated as an equivalent of Disabled.
<code>subPath</code>	<code>string</code>	Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).
<code>subPathExpr</code>	<code>string</code>	Expanded path within the volume from which the container's volume should be mounted. Behaves similarly to SubPath but environment variable references \$(VAR_NAME) are expanded using the container's environment. Defaults to "" (volume's root). SubPathExpr and SubPath are mutually exclusive.

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/stepactions`
 - `DELETE` : delete collection of StepAction
 - `GET` : list objects of kind StepAction
 - `POST` : create a new StepAction
- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/stepactions/{name}`
 - `DELETE` : delete the specified StepAction
 - `GET` : read the specified StepAction
 - `PATCH` : partially update the specified StepAction
 - `PUT` : replace the specified StepAction

/apis/tekton.dev/v1alpha1/namespaces/{namespace}/stepactions

HTTP method

`DELETE`

Description

delete collection of StepAction

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> ✓ schema
401 - Unauthorized	Empty

HTTP method

`GET`

Description

list objects of kind StepAction

HTTP responses

HTTP code	Response body
200 - OK	<code>StepActionList</code> schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new StepAction

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are

Parameter	Type	Description
		present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>StepAction</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>StepAction</code> schema
201 - Created	<code>StepAction</code> schema
202 - Accepted	<code>StepAction</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1alpha1/namespaces/{namespace}/stepactions/{name}

HTTP method

DELETE

Description

delete the specified StepAction

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result

Parameter	Type	Description
		in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
202 - Accepted	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

[GET](#)

Description

read the specified StepAction

HTTP responses

HTTP code	Response body
200 - OK	StepAction schema
401 - Unauthorized	Empty

HTTP method

[PATCH](#)

Description

partially update the specified StepAction

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive

Parameter	Type	Description
		will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>StepAction</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified StepAction

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>StepAction</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>StepAction</code> schema
201 - Created	<code>StepAction</code> schema
401 - Unauthorized	Empty





VerificationPolicy [tekton.dev/v1alpha1]

Description

VerificationPolicy defines the rules to verify Tekton resources. VerificationPolicy can config the mapping from resources to a list of public keys, so when verifying the resources we can use the corresponding public keys.

Type

object

Required

spec

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	string	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be

Property	Type	Description
		updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
<code>metadata</code>	<code>ObjectMeta</code> ↗	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the desired state of the VerificationPolicy.

.spec

Description

Spec holds the desired state of the VerificationPolicy.

Type

`object`

Required

`authorities`

`resources`

Property	Type	Description
<code>authorities</code>	<code>array</code>	Authorities defines the rules for validating signatures.
<code>mode</code>	<code>string</code>	Mode controls whether a failing policy will fail the taskrun/pipelinerun, or only log the warnings enforce - fail the taskrun/pipelinerun if verification fails (default) warn - don't fail the taskrun/pipelinerun if verification fails but log warnings

Property	Type	Description
<code>resources</code>	<code>array</code>	Resources defines the patterns of resources sources that should be subject to this policy. For example, we may want to apply this Policy from a certain GitHub repo. Then the ResourcesPattern should be valid regex. E.g. If using gitresolver, and we want to config keys from a certain git repo. <code>ResourcesPattern</code> can be <code>https://github.com/tektoncd/catalog.git</code> , we will use regex to filter out those resources.

.spec.authorities

Description

Authorities defines the rules for validating signatures.

Type

`array`

.spec.authorities[]

Description

The Authority block defines the keys for validating signatures.

Type

`object`

Required

`name`

Property	Type	Description
<code>key</code>	<code>object</code>	Key contains the public key to validate the resource.

Property	Type	Description
<code>name</code>	<code>string</code>	Name is the name for this authority.

`.spec.authorities[].key`

Description

Key contains the public key to validate the resource.

Type

`object`

Property	Type	Description
<code>data</code>	<code>string</code>	Data contains the inline public key.
<code>hashAlgorithm</code>	<code>string</code>	HashAlgorithm always defaults to sha256 if the algorithm hasn't been set.
<code>kms</code>	<code>string</code>	KMS contains the KMS url of the public key Supported formats d could be: gcpkms://projects/[PROJECT]/locations/[LOCATION]/keyRings/[KEYRING] For more examples please refer https://docs.sigstore.dev/cosign .
<code>secretRef</code>	<code>object</code>	SecretRef sets a reference to a secret with the key.

`.spec.authorities[].key.secretRef`

Description

SecretRef sets a reference to a secret with the key.

Type

object

Property	Type	Description
name	string	name is unique within a namespace to reference a secret resource.
namespace	string	namespace defines the space within which the secret name must be unique.

.spec.resources

Description

Resources defines the patterns of resources sources that should be subject to this policy. For example, we may want to apply this Policy from a certain GitHub repo. Then the ResourcesPattern should be valid regex. E.g. If using gitresolver, and we want to config keys from a certain git repo. `ResourcesPattern` can be `https://github.com/tektoncd/catalog.git`, we will use regex to filter out those resources.

Type

array

.spec.resources[]

Description

ResourcePattern defines the pattern of the resource source

Type

object

Required

pattern

Property	Type	Description
pattern	string	Pattern defines a resource pattern. Regex is created to filter resources based on <code>Pattern</code> Example patterns: GitHub resource: https://github.com/tektoncd/catalog.git , https://github.com/tektoncd/ * Bundle resource: gcr.io/tekton-releases/catalog/upstream/git-clone, gcr.io/tekton-releases/catalog/upstream/* Hub resource: https://artifacthub.io/ *

API Endpoints

The following API endpoints are available:

- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/verificationpolicies`
 - `DELETE` : delete collection of VerificationPolicy
 - `GET` : list objects of kind VerificationPolicy
 - `POST` : create a new VerificationPolicy
- `/apis/tekton.dev/v1alpha1/namespaces/{namespace}/verificationpolicies/{name}`
 - `DELETE` : delete the specified VerificationPolicy
 - `GET` : read the specified VerificationPolicy
 - `PATCH` : partially update the specified VerificationPolicy
 - `PUT` : replace the specified VerificationPolicy

`/apis/tekton.dev/v1alpha1/namespaces/{namespace}/verificationpolicies`

HTTP method

DELETE

Description

delete collection of VerificationPolicy

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind VerificationPolicy

HTTP responses

HTTP code	Response body
200 - OK	VerificationPolicyList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new VerificationPolicy

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing

Parameter	Type	Description
		of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>VerificationPolicy</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>VerificationPolicy</code> schema
201 - Created	<code>VerificationPolicy</code> schema
202 - Accepted	<code>VerificationPolicy</code> schema
401 - Unauthorized	Empty

/apis/tekton.dev/v1alpha1/namespaces/{namespace}/verificationpolicies/{name}

HTTP method

DELETE

Description

delete the specified VerificationPolicy

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code>  schema
202 - Accepted	<code>Status</code>  schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified VerificationPolicy

HTTP responses

HTTP code	Response body
200 - OK	<code>VerificationPolicy</code>  schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified VerificationPolicy

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>VerificationPolicy</code> schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified VerificationPolicy

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>VerificationPolicy</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>VerificationPolicy</code> schema
201 - Created	<code>VerificationPolicy</code> schema
401 - Unauthorized	Empty

ResolutionRequest

[resolution.tekton.dev/v1beta1]

Description

ResolutionRequest is an object for requesting the content of a Tekton resource like a pipeline.yaml.

Type

object

Specification

Property	Type	Description
apiVersion	string	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
kind	string	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info:

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
<code>metadata</code>	<code>ObjectMeta</code> ↗	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the information for the request part of the resource request.
<code>status</code>	<code>object</code>	Status communicates the state of the request and, ultimately, the content of the resolved resource.

.spec

Description

Spec holds the information for the request part of the resource request.

Type

`object`

Property	Type	Description
<code>params</code>	<code>object</code>	Parameters are the runtime attributes passed to the resolver to help it figure out how to resolve the resource being requested. For example: repo URL, commit SHA, path to file, the kind of authentication to leverage, etc.

.spec.params

Description

Parameters are the runtime attributes passed to the resolver to help it figure out how to resolve the resource being requested. For example: repo URL, commit SHA, path to file, the kind of authentication to leverage, etc.

Type

object

.status

Description

Status communicates the state of the request and, ultimately, the content of the resolved resource.

Type

object

Required

data

refSource

Property	Type	Description
annotations	object	Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.
conditions	array	Conditions the latest available observations of a resource's current state.

Property	Type	Description
<code>data</code>	<code>string</code>	Data is a string representation of the resolved content of the requested resource in-lined into the ResolutionRequest object.
<code>observedGeneration</code>	<code>integer</code>	ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.
<code>refSource</code>		RefSource is the source reference of the remote data that records where the remote file came from including the url, digest and the endpoint.

.status.annotations

Description

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

Type

`object`

.status.conditions

Description

Conditions the latest available observations of a resource's current state.

Type

`array`

.status.conditions[]

Description

Condition defines a readiness condition for a Knative resource. See: <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties>

Type

object

Required

status

type

Property	Type	Description
lastTransitionTime	string	LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this from creating equality.Semantic differences (all other things held constant).
message	string	A human readable message indicating details about the transition.
reason	string	The reason for the condition's last transition.
severity	string	Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.

Property	Type	Description
<code>status</code>	<code>string</code>	Status of the condition, one of True, False, Unknown.
<code>type</code>	<code>string</code>	Type of condition.

API Endpoints

The following API endpoints are available:

- `/apis/resolution.tekton.dev/v1alpha1/namespaces/{namespace}/resolutionrequests`
 - `DELETE` : delete collection of ResolutionRequest
 - `GET` : list objects of kind ResolutionRequest
 - `POST` : create a new ResolutionRequest
- `/apis/resolution.tekton.dev/v1alpha1/namespaces/{namespace}/resolutionrequests/{name}`
 - `DELETE` : delete the specified ResolutionRequest
 - `GET` : read the specified ResolutionRequest
 - `PATCH` : partially update the specified ResolutionRequest
 - `PUT` : replace the specified ResolutionRequest
- `/apis/resolution.tekton.dev/v1alpha1/namespaces/{namespace}/resolutionrequests/{name}/status`
 - `GET` : read status of the specified ResolutionRequest
 - `PATCH` : partially update status of the specified ResolutionRequest
 - `PUT` : replace status of the specified ResolutionRequest

/apis/resolution.tekton.dev/v1alpha1/namespaces/{namespace}/resolutionrequests

HTTP method

DELETE

Description

delete collection of ResolutionRequest

HTTP responses

HTTP code	Response body
200 - OK	Status schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ResolutionRequest

HTTP responses

HTTP code	Response body
200 - OK	ResolutionRequestList schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new ResolutionRequest

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized <code>dryRun</code> directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a <code>BadRequest</code> error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>ResolutionRequest</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>ResolutionRequest</code> schema
201 - Created	<code>ResolutionRequest</code> schema
202 - Accepted	<code>ResolutionRequest</code> schema

HTTP code	Response body
401 - Unauthorized	Empty

/apis/resolution.tekton.dev/v1alpha1/namespaces/{namespace}/resolutionrequests/{name}

HTTP method

DELETE

Description

delete the specified ResolutionRequest

Query parameters

Parameter	Type	Description
dryRun	string	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
202 - Accepted	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified ResolutionRequest

HTTP responses

HTTP code	Response body
200 - OK	<code>ResolutionRequest</code> schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ResolutionRequest

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>ResolutionRequest</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ResolutionRequest

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
body	ResolutionRequest schema	application/json formatted

HTTP responses

HTTP code	Response body
200 - OK	ResolutionRequest schema
201 - Created	ResolutionRequest schema
401 - Unauthorized	Empty

/apis/resolution.tekton.dev/v1alpha1/namespaces/{namespace}/resolutionrequests/{name}/status

HTTP method

GET

Description

read status of the specified ResolutionRequest

HTTP responses

HTTP code	Response body
200 - OK	ResolutionRequest schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified ResolutionRequest

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>ResolutionRequest</code> schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ResolutionRequest

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>ResolutionRequest</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>ResolutionRequest</code> schema
201 - Created	<code>ResolutionRequest</code> schema
401 - Unauthorized	Empty

ApprovalTask

[openshiftpipelines.org/v1alpha1]

Description

ApprovalTask is a "wait for manual approval" Task.

Type

object

Required

status

Specification

Property	Type	Description
<code>apiVersion</code>	<code>string</code>	APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources
<code>kind</code>	<code>string</code>	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be

Property	Type	Description
		updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
<code>metadata</code>	<code>ObjectMeta</code> ↗	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
<code>spec</code>	<code>object</code>	Spec holds the desired state of the TaskGroup from the client
<code>status</code>	<code>object</code>	

.spec

Description

Spec holds the desired state of the TaskGroup from the client

Type

`object`

Required

`approvers`

`numberOfApprovalsRequired`

Property	Type	Description
<code>approvers</code>	<code>array</code>	
<code>description</code>	<code>string</code>	
<code>numberOfApprovalsRequired</code>	<code>integer</code>	

`.spec.approvers`

Type

array

`.spec.approvers[]`

Type

object

Required

input

name

type

Property	Type	Description
<code>input</code>	<code>string</code>	
<code>message</code>	<code>string</code>	
<code>name</code>	<code>string</code>	
<code>type</code>	<code>string</code>	
<code>users</code>	<code>array</code>	

`.spec.approvers[].users`

Type

array

`.spec.approvers[].users[]`

Type

object

Required

input

name

Property	Type	Description
input	string	
name	string	

.status

Type

object

Required

state

Property	Type	Description
annotations	object	Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.
approvalsReceived	integer	ApprovalsReceived is the number of approvals received so far
approvalsRequired	integer	ApprovalsRequired is the number of approvals required for the task
approvers	array	
approversResponse	array	

Property	Type	Description
<code>conditions</code>	<code>array</code>	Conditions the latest available observations of a resource's current state.
<code>observedGeneration</code>	<code>integer</code>	ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.
<code>startTime</code>	<code>string</code>	StartTime is the time the build is actually started.
<code>state</code>	<code>string</code>	

.status.annotations

Description

Annotations is additional Status fields for the Resource to save some additional State as well as convey more information to the user. This is roughly akin to Annotations on any k8s resource, just the reconciler conveying richer information outwards.

Type

`object`

.status.approvers

Type

`array`

.status.approvers[]

Type

`string`

.status.approversResponse

Type

array

.status.approversResponse[]

Type

object

Required

name

response

type

Property	Type	Description
groupMembers	array	
message	string	
name	string	
response	string	
type	string	

.status.approversResponse[].groupMembers

Type

array

.status.approversResponse[].groupMembers[]

Type

object

Required

name

response

Property	Type	Description
message	string	
name	string	
response	string	

.status.conditions

Description

Conditions the latest available observations of a resource's current state.

Type

array

.status.conditions[]

Description

Condition defines a readiness condition for a Knative resource. See: <https://github.com/kubernetes/community/blob/master/contributors/devel/sig-architecture/api-conventions.md#typical-status-properties>

Type

object

Required

status

type

Property	Type	Description
lastTransitionTime	string	LastTransitionTime is the last time the condition transitioned from one status to another. We use VolatileTime in place of metav1.Time to exclude this

Property	Type	Description
		from creating equality.Semantic differences (all other things held constant).
<code>message</code>	<code>string</code>	A human readable message indicating details about the transition.
<code>reason</code>	<code>string</code>	The reason for the condition's last transition.
<code>severity</code>	<code>string</code>	Severity with which to treat failures of this type of condition. When this is not specified, it defaults to Error.
<code>status</code>	<code>string</code>	Status of the condition, one of True, False, Unknown.
<code>type</code>	<code>string</code>	Type of condition.

API Endpoints

The following API endpoints are available:

- `/apis/openshiftpipelines.org/v1alpha1/namespaces/{namespace}/approvaltasks`
 - `DELETE` : delete collection of ApprovalTask
 - `GET` : list objects of kind ApprovalTask
 - `POST` : create a new ApprovalTask

- `/apis/openshiftpipelines.org/v1alpha1/namespaces/{namespace}/approvaltasks/{name}`
 - **DELETE** : delete the specified ApprovalTask
 - **GET** : read the specified ApprovalTask
 - **PATCH** : partially update the specified ApprovalTask
 - **PUT** : replace the specified ApprovalTask
- `/apis/openshiftpipelines.org/v1alpha1/namespaces/{namespace}/approvaltasks/{name}/status`
 - **GET** : read status of the specified ApprovalTask
 - **PATCH** : partially update status of the specified ApprovalTask
 - **PUT** : replace status of the specified ApprovalTask

`/apis/openshiftpipelines.org/v1alpha1/namespaces/{namespace}/approvaltasks`

HTTP method

DELETE

Description

delete collection of ApprovalTask

HTTP responses

HTTP code	Response body
200 - OK	Status ↗ schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind ApprovalTask

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTaskList</code> schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a new ApprovalTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>ApprovalTask</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTask</code> schema
201 - Created	<code>ApprovalTask</code> schema
202 - Accepted	<code>ApprovalTask</code> schema
401 - Unauthorized	Empty

/apis/openshiftpipelines.org/v1alpha1/namespaces/{namespace}/approvaltasks/{name}

HTTP method

`DELETE`

Description

delete the specified ApprovalTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

HTTP code	Response body
200 - OK	<code>Status</code> schema
202 - Accepted	<code>Status</code> schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified ApprovalTask

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTask</code> schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified ApprovalTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore:

Parameter	Type	Description
		This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTask</code> schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified ApprovalTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

Parameter	Type	Description
<code>fieldValidation</code>	<code>string</code>	<code>fieldValidation</code> instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+. - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
<code>body</code>	<code>ApprovalTask</code> schema	<code>application/json</code> formatted

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTask</code> schema
201 - Created	<code>ApprovalTask</code> schema
401 - Unauthorized	Empty

`/apis/openshiftpipelines.org/v1alpha1/namespaces/{namespace}/approvaltasks/{name}/status`

HTTP method

GET

Description

read status of the specified ApprovalTask

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTask</code> schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified ApprovalTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default

Parameter	Type	Description
		in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	<code>ApprovalTask</code> schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified ApprovalTask

Query parameters

Parameter	Type	Description
<code>dryRun</code>	<code>string</code>	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed
<code>fieldValidation</code>	<code>string</code>	fieldValidation instructs the server on how to handle objects in the request (POST/PUT/PATCH) containing unknown or duplicate fields. Valid values are: - Ignore: This will ignore any unknown fields that are silently dropped from the object, and will ignore all but the last duplicate field that the decoder encounters. This is the default behavior prior to v1.23. - Warn: This will send a warning via the standard warning response header for each unknown field that is dropped from the object, and for each duplicate field that is encountered. The request

Parameter	Type	Description
		will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
body	ApprovalTask schema	application/json formatted

HTTP responses

HTTP code	Response body
200 - OK	ApprovalTask schema
201 - Created	ApprovalTask schema
401 - Unauthorized	Empty