



Pipelines

[Pipeline \[tekton.dev/v1\]](#)

[Task \[tekton.dev/v1\]](#)

[PipelineRun](#)

[TaskRun \[tekton.dev/v1\]](#)

[ClusterTask \[tekton.dev/v1\]](#)

[Run \[tekton](#)

[CustomRun \[tekton.dev/v1\]](#)

[StepAction \[tekton.dev/v1\]](#)

[Verification](#)

[ResolutionRequest \[resolution.tekton.dev/v1beta1\]](#)



Pipeline [tekton.dev/v1]

tekton.dev group

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

v1 version

▼ spec object

Spec holds the desired state of the Pipeline from the client

▼ description string

Description is a user-facing description of the pipeline that may be used to populate a UI.

▼ displayName string

DisplayName is a user-facing name of the pipeline that may be used to populate a UI.

▼ finally []object

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

▼ description string

Description is the description of this task within the context of a Pipeline. This description may be used to populate a UI.

▼ displayName `string`

DisplayName is the display name of this task within the context of a Pipeline. This display name may be used to populate a UI.

▼ matrix `object`

Matrix declares parameters used to fan out this task.

▼ include `[]object`

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

▼ name `string`

Name the specified combination

▼ params `[]object`

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

▼ name `string` required**▼ value** required**▼ params** `[]object`

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

▼ **name** **string** **required**

▼ **value** **required**

▼ **name** **string**

Name is the name of this task within the context of a Pipeline. Name is used as a coordinate with the **from** and **runAfter** fields to establish the execution order of tasks relative to one another.

▼ **onError** **string**

OnError defines the exiting behavior of a PipelineRun on error can be set to [continue | stopAndFail]

▼ **params** **[]object**

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

▼ **name** **string** **required**

▼ **value** **required**

▼ **pipelineRef** **object**

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

▼ apiVersion `string`

API version of the referent

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ name `string` required**▼ value** required**▼ resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ pipelineSpec

PipelineSpec is a specification of a pipeline Note: PipelineSpec is in preview mode and not yet supported Specifying PipelineSpec can be disabled by setting `disable-inline-spec` feature flag. See Pipeline.spec (API version: tekton.dev/v1)

▼ retries `integer`

Retries represents how many times this task should be retried in case of task failure: ConditionSucceeded set to False

▼ runAfter `[]string`

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

▼ taskRef `object`

TaskRef is a reference to a task definition.

▼ 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:

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

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ **resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ **taskSpec**

TaskSpec is a specification of a task Specifying TaskSpec can be disabled by setting `disable-inline-spec` feature flag. See Task.spec (API version: tekton.dev/v1)

▼ **timeout** `string`

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> ↗

▼ **when** `[]object`

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

▼ **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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workspaces `[]object`

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

▼ name `string` required

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

▼ params `[]object`

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources).

Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

▼ default

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ properties `object`

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

▼ type `string`

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ results `[]object`

PipelineResult used to describe the results of a pipeline

▼ description `string`

Description is a human-readable description of the result

▼ name `string` required

Name the given name

▼ type `string`

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.

▼ value required

Value the expression used to retrieve the value

▼ tasks `[]object`

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

▼ description `string`

Description is the description of this task within the context of a Pipeline. This description may be used to populate a UI.

▼ displayName `string`

DisplayName is the display name of this task within the context of a Pipeline. This display name may be used to populate a UI.

▼ matrix `object`

Matrix declares parameters used to fan out this task.

▼ include `[]object`

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

▼ name `string`

Name the specified combination

▼ params `[]object`

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

▼ name `string` required**▼ value** required

▼ params `[]object`

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

▼ name `string` `required`**▼ value** `required`**▼ name** `string`

Name is the name of this task within the context of a Pipeline. Name is used as a coordinate with the `from` and `runAfter` fields to establish the execution order of tasks relative to one another.

▼ onError `string`

OnError defines the exiting behavior of a PipelineRun on error can be set to [`continue` | `stopAndFail`]

▼ params `[]object`

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

▼ name `string` `required`**▼ value** `required`

▼ pipelineRef `object`

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

▼ apiVersion `string`

API version of the referent

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ name `string` required

▼ value required

▼ resolver `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ pipelineSpec

PipelineSpec is a specification of a pipeline Note: PipelineSpec is in preview mode and not yet supported Specifying PipelineSpec can be disabled by setting `disable-inline-spec` feature flag. See Pipeline.spec (API version: tekton.dev/v1)

▼ retries `integer`

Retries represents how many times this task should be retried in case of task failure: ConditionSucceeded set to False

▼ runAfter `[]string`

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

▼ taskRef `object`

TaskRef is a reference to a task definition.

▼ 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:

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

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ name `string` required**▼ value** required**▼ resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ taskSpec

TaskSpec is a specification of a task Specifying TaskSpec can be disabled by setting `disable-inline-spec` feature flag. See Task.spec (API version: tekton.dev/v1)

▼ timeout `string`

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> ↗

▼ when `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workspaces `[]object`

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

▼ name `string` required

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

▼ workspaces `[]object`

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

▼ description `string`

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.

▼ name `string` required

Name is the name of a workspace to be provided by a PipelineRun.

▼ optional `boolean`

Optional marks a Workspace as not being required in PipelineRuns. By default this field is false and so declared workspaces are required.

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

Deprecated: Please use v1.Pipeline instead.

v1beta1

version

▼ spec **object**

Spec holds the desired state of the Pipeline from the client

▼ description **string**

Description is a user-facing description of the pipeline that may be used to populate a UI.

▼ displayName **string**

DisplayName is a user-facing name of the pipeline that may be used to populate a UI.

▼ finally **[]object**

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

▼ description **string**

Description is the description of this task within the context of a Pipeline. This description may be used to populate a UI.

▼ displayName **string**

DisplayName is the display name of this task within the context of a Pipeline. This display name may be used to populate a UI.

▼ matrix **object**

Matrix declares parameters used to fan out this task.

▼ include **[]object**

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

▼ name **string**

Name the specified combination

▼ params **[]object**

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

▼ name **string** required

▼ value required

▼ params **[]object**

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

▼ name **string** required

▼ **value** required

▼ **name** string

Name is the name of this task within the context of a Pipeline. Name is used as a coordinate with the from and runAfter fields to establish the execution order of tasks relative to one another.

▼ **onError** string

OnError defines the exiting behavior of a PipelineRun on error can be set to [continue | stopAndFail]

▼ **params** []object

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

▼ **name** string required

▼ **value** required

▼ **pipelineRef** object

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

▼ **apiVersion** string

API version of the referent

▼ **bundle** `string`

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.

▼ **name** `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ **params** `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ **resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ **pipelineSpec**

PipelineSpec is a specification of a pipeline Note: PipelineSpec is in preview mode and not yet supported Specifying PipelineSpec can be disabled by setting `disable-inline-spec` feature flag. See Pipeline.spec (API version: tekton.dev/v1beta1)

▼ resources `object`

Deprecated: Unused, preserved only for backwards compatibility

▼ inputs `[]object`

PipelineTaskInputResource maps the name of a declared PipelineResource input dependency in a Task to the resource in the Pipeline's DeclaredPipelineResources that should be used. This input may come from a previous task.

Deprecated: Unused, preserved only for backwards compatibility

▼ from `[]string`

From is the list of PipelineTask names that the resource has to come from. (Implies an ordering in the execution graph.)

▼ name `string` required

Name is the name of the PipelineResource as declared by the Task.

▼ resource `string` required

Resource is the name of the DeclaredPipelineResource to use.

▼ outputs `[]object`

PipelineTaskOutputResource maps the name of a declared PipelineResource output dependency in a Task to the resource in the Pipeline's DeclaredPipelineResources that should be used.

Deprecated: Unused, preserved only for backwards compatibility

▼ **name** `string` required

Name is the name of the PipelineResource as declared by the Task.

▼ **resource** `string` required

Resource is the name of the DeclaredPipelineResource to use.

▼ **retries** `integer`

Retries represents how many times this task should be retried in case of task failure: ConditionSucceeded set to False

▼ **runAfter** `[]string`

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

▼ **taskRef** `object`

TaskRef is a reference to a task definition.

▼ **apiVersion** `string`

API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task

▼ bundle `string`

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.

▼ kind `string`

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

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ name `string` required**▼ value** required**▼ resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ taskSpec

TaskSpec is a specification of a task Specifying TaskSpec can be disabled by setting `disable-inline-spec` feature flag. See Task.spec (API version: tekton.dev/v1beta1)

▼ timeout `string`

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> ↗

▼ when `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workspaces `[]object`

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

▼ name `string` required

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

▼ params `[]object`

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

▼ default

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ properties `object`

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

▼ type `string`

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ resources `[]object`

PipelineDeclaredResource is used by a Pipeline to declare the types of the PipelineResources that it will required to run and names which can be used to refer to these PipelineResources in PipelineTaskResourceBindings.

Deprecated: Unused, preserved only for backwards compatibility

▼ **name** `string` required

Name is the name that will be used by the Pipeline to refer to this resource. It does not directly correspond to the name of any PipelineResources Task inputs or outputs, and it does not correspond to the actual names of the PipelineResources that will be bound in the PipelineRun.

▼ **optional** `boolean`

Optional declares the resource as optional. optional: true - the resource is considered optional optional: false - the resource is considered required (default/equivalent of not specifying it)

▼ **type** `string` required

Type is the type of the PipelineResource.

▼ **results** `[]object`

PipelineResult used to describe the results of a pipeline

▼ **description** `string`

Description is a human-readable description of the result

▼ **name** `string` required

Name the given name

▼ **type** `string`

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.

▼ value required

Value the expression used to retrieve the value

▼ tasks []object

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

▼ description string

Description is the description of this task within the context of a Pipeline. This description may be used to populate a UI.

▼ displayName string

DisplayName is the display name of this task within the context of a Pipeline. This display name may be used to populate a UI.

▼ matrix object

Matrix declares parameters used to fan out this task.

▼ include []object

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

▼ name string

Name the specified combination

▼ **params** `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ **params** `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ **name** `string`

Name is the name of this task within the context of a Pipeline. Name is used as a coordinate with the `from` and `runAfter` fields to establish the execution order of tasks relative to one another.

▼ **onError** `string`

OnError defines the exiting behavior of a PipelineRun on error can be set to [continue | stopAndFail]

▼ params `[]object`

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

▼ name `string` required

▼ value required

▼ pipelineRef `object`

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

▼ apiVersion `string`

API version of the referent

▼ bundle `string`

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.

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params []object

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

▼ name string required

▼ value required

▼ resolver string

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ pipelineSpec

PipelineSpec is a specification of a pipeline Note: PipelineSpec is in preview mode and not yet supported Specifying PipelineSpec can be disabled by setting

`disable-inline-spec` feature flag. See Pipeline.spec (API version: tekton.dev/v1beta1)

▼ resources object

Deprecated: Unused, preserved only for backwards compatibility

▼ inputs []object

PipelineTaskInputResource maps the name of a declared PipelineResource input dependency in a Task to the resource in the

Pipeline's DeclaredPipelineResources that should be used. This input may come from a previous task.

Deprecated: Unused, preserved only for backwards compatibility

▼ from `[]string`

From is the list of PipelineTask names that the resource has to come from. (Implies an ordering in the execution graph.)

▼ name `string` required

Name is the name of the PipelineResource as declared by the Task.

▼ resource `string` required

Resource is the name of the DeclaredPipelineResource to use.

▼ outputs `[]object`

PipelineTaskOutputResource maps the name of a declared PipelineResource output dependency in a Task to the resource in the Pipeline's DeclaredPipelineResources that should be used.

Deprecated: Unused, preserved only for backwards compatibility

▼ name `string` required

Name is the name of the PipelineResource as declared by the Task.

▼ resource `string` required

Resource is the name of the DeclaredPipelineResource to use.

▼ retries `integer`

Retries represents how many times this task should be retried in case of task failure: ConditionSucceeded set to False

▼ runAfter `[]string`

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

▼ taskRef `object`

TaskRef is a reference to a task definition.

▼ apiVersion `string`

API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task

▼ bundle `string`

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.

▼ kind `string`

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

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ name `string` required**▼ value** required**▼ resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ taskSpec

TaskSpec is a specification of a task Specifying TaskSpec can be disabled by setting `disable-inline-spec` feature flag. See Task.spec (API version: tekton.dev/v1beta1)

▼ timeout `string`

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> ↗

▼ when `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workspaces `[]object`

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

▼ **name** `string` required

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

▼ **workspaces** `[]object`

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

▼ **description** `string`

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.

▼ **name** `string` required

Name is the name of a workspace to be provided by a PipelineRun.

▼ optional `boolean`

Optional marks a Workspace as not being required in PipelineRuns. By default this field is false and so declared workspaces are required.



Task [tekton.dev/v1]

tekton.dev group

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.

v1 version

▼ spec object

Spec holds the desired state of the Task from the client

▼ description string

Description is a user-facing description of the task that may be used to populate a UI.

▼ displayName string

DisplayName is a user-facing name of the task that may be used to populate a UI.

▼ params []object

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

▼ default

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ properties `object`

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

▼ type `string`

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ results `[]object`

TaskResult used to describe the results of a task

▼ description `string`

Description is a human-readable description of the result

▼ name `string` required

Name the given name

▼ properties `object`

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

▼ 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 expression used to retrieve the value of the result from an underlying Step.

▼ sidecars `[]object`

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

▼ args `[]string`

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** []string

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> ↗

▼ **computeResources** object

ComputeResources required by this Sidecar. Cannot be updated. More info:

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

▼ **claims** []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ **name** string required

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.

▼ 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/> ↗

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** **required**

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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.

▼ apiVersion string

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath string required

Path of the field to select in the specified API version.

▼ resourceFieldRef 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.

▼ containerName string

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource string required

Required: resource to select

▼ secretKeyRef object

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

▼ **key** `string` required

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

▼ **envFrom** `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ **configMapRef** `object`

The ConfigMap to select from

▼ **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 must be defined

▼ prefix **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef **object**

The Secret to select from

▼ 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

▼ image **string**

Image reference name. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ **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#updating-images> ↗

▼ **lifecycle** `object`

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

▼ **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> ↗

▼ **exec** `object`

Exec specifies a command to execute in the container.

▼ **command** `[]string`

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **sleep** `object`

Sleep represents a duration that the container should sleep.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ 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> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

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

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ sleep **object**

Sleep represents a duration that the container should sleep.

▼ seconds **integer** **required**

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

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.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ livenessProbe **object**

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>



▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** 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> ↗

▼ **periodSeconds** integer

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** object

TCPSocket specifies a connection to a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** `required`

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ name **string** required

Name of the Sidecar specified as a DNS_LABEL. Each Sidecar in a Task must have a unique name (DNS_LABEL). Cannot be updated.

▼ ports **[]object**

ContainerPort represents a network port in a single container.

▼ containerPort **integer** required

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.

▼ protocol **string**

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ 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/pod-lifecycle#container-probes>



▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet `object`

HTTPGet specifies an HTTP GET request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** `required`

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies a connection to a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ restartPolicy `string`

RestartPolicy refers to kubernetes RestartPolicy. It can only be set for an initContainer and must have its policy set to "Always". It is currently left optional to help support Kubernetes versions prior to 1.29 when this feature was introduced.

▼ script `string`

Script is the contents of an executable file to execute.

If Script is not empty, the Step cannot have an Command or Args.

▼ securityContext `object`

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/> ↗

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ appArmorProfile **object**

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.

▼ localhostProfile **string**

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".

▼ type **string** required

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.

▼ 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.

▼ add **[]string**

Added capabilities

▼ drop **[]string**

Removed capabilities

▼ **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 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 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.

▼ 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.

▼ 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.

▼ 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** **required**

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.

▼ windowsOptions **object**

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.

▼ 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.

▼ 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 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.

▼ startupProbe `object`

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>



▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet `object`

HTTPGet specifies an HTTP GET request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** `required`

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies a connection to a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ **stdin** **boolean**

Whether this Sidecar should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the Sidecar will always result in EOF. Default is false.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on Sidecar start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the Sidecar is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ **terminationMessagePath** **string**

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, such as an assertion failure message. Will be truncated by the node 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.

▼ **terminationMessagePolicy** **string**

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 Sidecar

log output if the termination message file is empty and the Sidecar exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ tty `boolean`

Whether this Sidecar should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices `[]object`

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

▼ devicePath `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ name `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain ':'.

▼ mountPropagation `string`

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` required

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.

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.

▼ **subPath** `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ **subPathExpr** `string`

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.

▼ **workingDir** `string`

Sidecar'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.

▼ **workspaces** `[]object`

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

▼ **mountPath** `string` required

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.

▼ **name** `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ stepTemplate **object**

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

▼ args **[]string**

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> ↗

▼ command **[]string**

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> ↗

▼ computeResources **object**

ComputeResources required by this Step. Cannot be updated. More info:

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ **name** **string** required

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.

▼ **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/> ↗

▼ **env** **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef `object`

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`.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef `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.

▼ containerName `string`

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource `string` required

Required: resource to select

▼ secretKeyRef `object`

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

▼ key `string` required

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

▼ envFrom `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef `object`

The ConfigMap to select from

▼ 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 must be defined

▼ prefix `string`

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef `object`

The Secret to select from

▼ 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

▼ image `string`

Image reference name. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ 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#updating-images> ↗

▼ 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/> ↗

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the

no_new_privs 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.

▼ appArmorProfile object

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.

▼ localhostProfile string

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".

▼ type string required

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.

▼ 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.

▼ add []string

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **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 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 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.

▼ 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.

▼ 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.

▼ 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` required

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.

▼ windowsOptions `object`

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.

▼ **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.

▼ **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 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.

▼ volumeDevices `[]object`

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

▼ devicePath `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ name `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain ':'.

▼ mountPropagation `string`

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` required

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.

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.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ **workingDir** `string`

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.

▼ **steps** `[]object`

Step runs a subcomponent of a Task

▼ **args** `[]string`

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> ↗

▼ **command** `[]string`

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> ↗

▼ computeResources **object**

ComputeResources required by this Step. Cannot be updated. More info:

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** required

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.

▼ 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/> ↗

▼ env []object

EnvVar represents an environment variable present in a Container.

▼ name string required

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.

▼ configMapKeyRef object

Selects a key of a ConfigMap.

▼ key **string** **required**

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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** **required**

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** *required*

Required: resource to select

▼ secretKeyRef **object**

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

▼ key **string** *required*

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

▼ envFrom `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef `object`

The ConfigMap to select from

▼ 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 must be defined

▼ **prefix** `string`

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** `object`

The Secret to select from

▼ **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

▼ **image** `string`

Docker image name. More info: <https://kubernetes.io/docs/concepts/containers/images> ↗

▼ **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#updating-images> ↗

▼ **name** `string` required

Name of the Step specified as a DNS_LABEL. Each Step in a Task must have a unique name.

▼ **onError** `string`

OnError defines the exiting behavior of a container on error can be set to [continue | stopAndFail]

▼ **params** `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ **ref** `object`

Contains the reference to an existing StepAction.

▼ **name** `string`

Name of the referenced step

▼ params `[]object`

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

▼ name `string` `required`**▼ value** `required`**▼ resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ results `[]object`

StepResult used to describe the Results of a Step.

▼ description `string`

Description is a human-readable description of the result

▼ name `string` `required`

Name the given name

▼ properties `object`

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

▼ type `string`

The possible types are 'string', 'array', and 'object', with 'string' as the default.

▼ 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/> ↗

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ appArmorProfile `object`

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.

▼ localhostProfile `string`

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".

▼ type `string` required

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.

▼ 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.

▼ add `[]string`

Added capabilities

▼ drop `[]string`

Removed capabilities

▼ 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 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 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.

▼ 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.

▼ **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.

▼ **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** **required**

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.

▼ **windowsOptions** **object**

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.

▼ 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.

▼ 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 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.

▼ stderrConfig `object`

Stores configuration for the stderr stream of the step.

▼ path **string**

Path to duplicate stdout stream to on container's local filesystem.

▼ stdoutConfig **object**

Stores configuration for the stdout stream of the step.

▼ path **string**

Path to duplicate stdout stream to on container's local filesystem.

▼ timeout **string**

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> ↗

▼ volumeDevices **[]object**

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

▼ devicePath **string** **required**

devicePath is the path inside of the container that the device will be mapped to.

▼ name **string** **required**

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []object

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath string required

Path within the container at which the volume should be mounted. Must not contain '..'.

▼ mountPropagation string

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 required

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.

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.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ when `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workingDir `string`

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.

▼ workspaces `[]object`

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

▼ mountPath `string` required

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.

▼ name `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ volumes

Volumes is a collection of volumes that are available to mount into the steps of the build.

See Pod.spec.volumes (API version: v1)

▼ workspaces `[]object`

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

▼ description `string`

Description is an optional human readable description of this volume.

▼ mountPath `string`

MountPath overrides the directory that the volume will be made available at.

▼ name `string` required

Name is the name by which you can bind the volume at runtime.

▼ optional `boolean`

Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.

▼ readOnly `boolean`

ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.

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.

Deprecated: Please use v1.Task instead.

v1beta1

version

▼ spec **object**

Spec holds the desired state of the Task from the client

▼ description **string**

Description is a user-facing description of the task that may be used to populate a UI.

▼ displayName **string**

DisplayName is a user-facing name of the task that may be used to populate a UI.

▼ params **[]object**

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources).

Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

▼ default

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.

▼ description **string**

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ properties `object`

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

▼ type `string`

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ resources `object`

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

▼ inputs `[]object`

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

▼ **description** `string`

Description is a user-facing description of the declared resource that may be used to populate a UI.

▼ **name** `string` `required`

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.

▼ **optional** `boolean`

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)

▼ **targetPath** `string`

TargetPath is the path in workspace directory where the resource will be copied.

▼ **type** `string` `required`

Type is the type of this resource;

▼ **outputs** `[]object`

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

▼ **description** `string`

Description is a user-facing description of the declared resource that may be used to populate a UI.

▼ **name** `string` required

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.

▼ **optional** `boolean`

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)

▼ **targetPath** `string`

TargetPath is the path in workspace directory where the resource will be copied.

▼ **type** `string` required

Type is the type of this resource;

▼ results `[]object`

TaskResult used to describe the results of a task

▼ description `string`

Description is a human-readable description of the result

▼ name `string` required

Name the given name

▼ properties `object`

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

▼ 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 expression used to retrieve the value of the result from an underlying Step.

▼ sidecars `[]object`

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

▼ args `[]string`

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> ↗

▼ command `[]string`

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> ↗

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** required

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ **envFrom** `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ **configMapRef** **object**

The ConfigMap to select from

▼ **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 must be defined

▼ **prefix** **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** **object**

The Secret to select from

▼ **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

▼ 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, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ lifecycle `object`

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

▼ 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> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ httpGet `object`

HTTPGet specifies an HTTP GET request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port **required**

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.

▼ sleep **object**

Sleep represents a duration that the container should sleep.

▼ seconds **integer** **required**

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

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.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ 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> ↗

▼ exec object

Exec specifies a command to execute in the container.

▼ command []string

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** **required**

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

▼ value **string** **required**

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port **required**

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.

▼ **sleep** `object`

Sleep represents a duration that the container should sleep.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ livenessProbe **object**

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>



▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ initialDelaySeconds **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> ↗

▼ periodSeconds **integer**

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ successThreshold **integer**

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies a connection to a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **name** `string` required

Name of the Sidecar specified as a DNS_LABEL. Each Sidecar in a Task must have a unique name (DNS_LABEL). Cannot be updated.

▼ **ports** `[]object`

ContainerPort represents a network port in a single container.

▼ **containerPort** `integer` required

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.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ 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/pod-lifecycle#container-probes>



▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** **required**

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** **required**

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

▼ **value** **string** required

The header field value

▼ **path** **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** **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> ↗

▼ **periodSeconds** **integer**

How often (in seconds) to perform the probe. Default to 10 seconds.
Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.
Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies a connection to a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** `required`

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill

signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ resources object

Compute Resources required by this Sidecar. Cannot be updated. More info:

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

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ 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/> ↗

▼ restartPolicy **string**

RestartPolicy refers to kubernetes RestartPolicy. It can only be set for an initContainer and must have it's policy set to "Always". It is currently left optional to help support Kubernetes versions prior to 1.29 when this feature was introduced.

▼ script **string**

Script is the contents of an executable file to execute.

If Script is not empty, the Step cannot have an Command or Args.

▼ securityContext **object**

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/> ↗

▼ allowPrivilegeEscalation boolean

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ appArmorProfile object

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.

▼ localhostProfile string

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".

▼ type string required

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.

▼ 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.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **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 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 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.

▼ 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.

▼ **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.

▼ **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` `required`

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.

▼ windowsOptions **object**

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.

▼ 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.

▼ 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 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.

▼ startupProbe `object`

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>

**▼ exec** `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ **port** **integer** required

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.

▼ **httpGet** **object**

HTTPGet specifies an HTTP GET request to perform.

▼ **host** **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ initialDelaySeconds `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> ↗

▼ periodSeconds `integer`

How often (in seconds) to perform the probe. Default to 10 seconds.

Minimum value is 1.

▼ successThreshold `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket `object`

TCPSocket specifies a connection to a TCP port.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port `required`

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.

▼ terminationGracePeriodSeconds `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ stdin boolean

Whether this Sidecar should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the Sidecar will always result in EOF. Default is false.

▼ stdinOnce boolean

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on Sidecar start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the Sidecar is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath string

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, such as an assertion failure message. Will be truncated by the

node 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.

▼ **terminationMessagePolicy** `string`

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 Sidecar log output if the termination message file is empty and the Sidecar exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ **tty** `boolean`

Whether this Sidecar should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ **volumeDevices** `[]object`

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

▼ **devicePath** `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ **name** `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ **volumeMounts** `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath **string** required

Path within the container at which the volume should be mounted. Must not contain '.'.

▼ mountPropagation **string**

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** required

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.

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`.

▼ **subPath** `string`

Path within the volume from which the container's volume should be mounted. Defaults to `""` (volume's root).

▼ **subPathExpr** `string`

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.

▼ **workingDir** `string`

Sidecar'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.

▼ **workspaces** `[]object`

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

▼ **mountPath** `string` required

`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`.

▼ name **string** required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ stepTemplate **object**

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

▼ args **[]string**

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> ↗

▼ command **[]string**

Entrypoint array. Not executed within a shell. The docker 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> ↗

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** **required**

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** **required**

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

▼ fieldRef `object`

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.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ **envFrom** **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ **configMapRef** **object**

The ConfigMap to select from

▼ **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 must be defined

▼ **prefix** **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef **object**

The Secret to select from

▼ 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

▼ 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 like 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#updating-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.

▼ **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> ↗

▼ **exec** **object**

Exec specifies a command to execute in the container.

▼ **command** **[]string**

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.

▼ **httpGet** **object**

HTTPGet specifies an HTTP GET request to perform.

▼ **host** **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ sleep `object`

Sleep represents a duration that the container should sleep.

▼ **seconds** **integer** required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** **object**

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.

▼ **host** **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ **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> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ **value** **string** required

The header field value

▼ **path** **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **sleep** **object**

Sleep represents a duration that the container should sleep.

▼ **seconds** **integer** required

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

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.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ 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/pod-lifecycle#container-probes>



Deprecated: This field will be removed in a future release.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies a GRPC HealthCheckRequest.

▼ **port** `integer` required

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.

▼ **httpGet** `object`

HTTPGet specifies an HTTP GET request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ initialDelaySeconds **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> ↗

▼ periodSeconds **integer**

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ successThreshold **integer**

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies a connection to a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ **terminationGracePeriodSeconds** integer

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **name** string required

Default name for each Step specified as a DNS_LABEL. Each Step in a Task must have a unique name. Cannot be updated.

Deprecated: This field will be removed in a future release.

▼ **ports** []object

ContainerPort represents a network port in a single container.

▼ **containerPort** integer required

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.

▼ **protocol** `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ **readinessProbe** `object`

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.

▼ **exec** `object`

Exec specifies a command to execute in the container.

▼ **command** `[]string`

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.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies a GRPC HealthCheckRequest.

▼ **port** `integer` required

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies a connection to a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ terminationGracePeriodSeconds integer

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ resources object

Compute Resources required by this Step. Cannot be updated. More info:

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** **required**

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.

▼ 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/> ↗

▼ 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/> ↗

▼ allowPrivilegeEscalation **boolean**

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs 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.

▼ appArmorProfile **object**

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.

▼ localhostProfile **string**

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".

▼ type **string** required

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.

▼ 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.

▼ add `[]string`

Added capabilities

▼ drop `[]string`

Removed capabilities

▼ 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 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 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.

▼ **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.

▼ **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.

▼ **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` required

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.

▼ **windowsOptions** `object`

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.

▼ **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.

▼ **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 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.

▼ startupProbe `object`

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.

▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies a GRPC HealthCheckRequest.

▼ **port** `integer` required

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.

▼ **httpGet** `object`

HTTPGet specifies an HTTP GET request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ **initialDelaySeconds** **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> ↗

▼ **periodSeconds** **integer**

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** **integer**

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** **object**

TCP socket specifies a connection to a TCP port.

▼ **host** **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** **required**

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.

▼ **terminationGracePeriodSeconds** **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **stdin** **boolean**

Whether this Step should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the Step will always result in EOF. Default is false.

Deprecated: This field will be removed in a future release.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true 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, and then 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

Deprecated: This field will be removed in a future release.

▼ **terminationMessagePath** `string`

Deprecated: This field will be removed in a future release and cannot be meaningfully used.

▼ **terminationMessagePolicy** `string`

Deprecated: This field will be removed in a future release and cannot be meaningfully used.

▼ **tty** `boolean`

Whether this Step should allocate a DeprecatedTTY for itself, also requires 'stdin' to be true. Default is false.

Deprecated: This field will be removed in a future release.

▼ **volumeDevices** `[]object`

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

▼ **devicePath** `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ name `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain '..'.

▼ mountPropagation `string`

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` required

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.

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`.

▼ `subPath` `string`

Path within the volume from which the container's volume should be mounted. Defaults to `""` (volume's root).

▼ `subPathExpr` `string`

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.

▼ `workingDir` `string`

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.

▼ steps `[]object`

Step runs a subcomponent of a Task

▼ args `[]string`

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> ↗

▼ command `[]string`

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> ↗

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** required

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** **string** required

Required: resource to select

▼ **secretKeyRef** **object**

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

▼ **key** **string** required

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

▼ **envFrom** **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ **configMapRef** **object**

The ConfigMap to select from

▼ **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 must be defined

▼ **prefix** **string**

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** **object**

The Secret to select from

▼ **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

▼ image `string`

Image reference name to run for this Step. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ 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#updating-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.

▼ 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> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ **command** `[]string`

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.

▼ **httpGet** `object`

HTTPGet specifies an HTTP GET request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

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

▼ **value** `string` required

The header field value

▼ **path** **string**

Path to access on the HTTP server.

▼ **port** **required**

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.

▼ **sleep** **object**

Sleep represents a duration that the container should sleep.

▼ **seconds** **integer** **required**

Seconds is the number of seconds to sleep.

▼ **tcpSocket** **object**

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.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port `required`

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.

▼ 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> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **sleep** `object`

Sleep represents a duration that the container should sleep.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ livenessProbe **object**

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.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value string required

The header field value

▼ path string

Path to access on the HTTP server.

▼ port required

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.

▼ initialDelaySeconds 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> ↗

▼ periodSeconds integer

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ successThreshold integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies a connection to a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ name string required

Name of the Step specified as a DNS_LABEL. Each Step in a Task must have a unique name.

▼ onError string

OnError defines the exiting behavior of a container on error can be set to [continue | stopAndFail]

▼ params []object

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

▼ name string required**▼ value** required**▼ ports** []object

ContainerPort represents a network port in a single container.

▼ containerPort integer required

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.

▼ **protocol** `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ **readinessProbe** `object`

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.

▼ **exec** `object`

Exec specifies a command to execute in the container.

▼ **command** `[]string`

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.

▼ **failureThreshold** `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** `object`

GRPC specifies a GRPC HealthCheckRequest.

▼ **port** `integer` required

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies a connection to a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ terminationGracePeriodSeconds integer

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ ref object

Contains the reference to an existing StepAction.

▼ name string

Name of the referenced step

▼ params []object

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

▼ name string required**▼ value** required**▼ resolver** string

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ resources object

Compute Resources required by this Step. Cannot be updated. More info:

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

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ limits `object`

Limits describes the maximum amount of compute resources allowed.

More info: [https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗](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/ ↗](https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/)

▼ results `[]object`

StepResult used to describe the Results of a Step.

▼ description `string`

Description is a human-readable description of the result

▼ name `string` required

Name the given name

▼ properties **object**

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

▼ type **string**

The possible types are 'string', 'array', and 'object', with 'string' as the default.

▼ 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/> ↗

▼ allowPrivilegeEscalation **boolean**

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ appArmorProfile **object**

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.

▼ localhostProfile **string**

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".

▼ type **string** required

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.

▼ 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.

▼ add **[]string**

Added capabilities

▼ drop **[]string**

Removed capabilities

▼ **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 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 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.

▼ 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.

▼ 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.

▼ 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** **required**

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.

▼ windowsOptions **object**

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.

▼ gmsaCredentialSpec `string`

GMSACredentialSpec is where the GMSA admission webhook ([https://github.com/kubernetes-sigs/windows-gmsa ↗](https://github.com/kubernetes-sigs/windows-gmsa)) inlines the contents of the GMSA credential spec named by the GMSACredentialSpecName 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 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.

▼ startupProbe `object`

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.

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet `object`

HTTPGet specifies an HTTP GET request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** `required`

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies a connection to a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ stderrConfig **object**

Stores configuration for the stderr stream of the step.

▼ path **string**

Path to duplicate stdout stream to on container's local filesystem.

▼ stdin **boolean**

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is false.

Deprecated: This field will be removed in a future release.

▼ stdinOnce **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true 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, and then 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

Deprecated: This field will be removed in a future release.

▼ stdoutConfig **object**

Stores configuration for the stdout stream of the step.

▼ path **string**

Path to duplicate stdout stream to on container's local filesystem.

▼ terminationMessagePath **string**

Deprecated: This field will be removed in a future release and can't be meaningfully used.

▼ terminationMessagePolicy **string**

Deprecated: This field will be removed in a future release and can't be meaningfully used.

▼ timeout **string**

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> ↗

▼ tty **boolean**

Whether this container should allocate a DeprecatedTTY for itself, also requires 'stdin' to be true. Default is false.

Deprecated: This field will be removed in a future release.

▼ volumeDevices **[]object**

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

▼ **devicePath** `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ **name** `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ **volumeMounts** `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ **mountPath** `string` required

Path within the container at which the volume should be mounted. Must not contain ':'.

▼ **mountPropagation** `string`

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` required

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.

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.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ when `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workingDir `string`

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.

▼ workspaces `[]object`

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

▼ mountPath `string` required

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.

▼ name `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ volumes

Volumes is a collection of volumes that are available to mount into the steps of the build.
See Pod.spec.volumes (API version: v1)

▼ workspaces `[]object`

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

▼ description `string`

Description is an optional human readable description of this volume.

▼ mountPath `string`

MountPath overrides the directory that the volume will be made available at.

▼ name **string** **required**

Name is the name by which you can bind the volume at runtime.

▼ optional **boolean**

Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.

▼ readOnly **boolean**

ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.



PipelineRun [tekton.dev/v1]

tekton.dev group

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.

v1 version

▼ spec object

PipelineRunSpec defines the desired state of PipelineRun

▼ params []object

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

▼ name string required

▼ value required

▼ pipelineRef object

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

▼ apiVersion string

API version of the referent

▼ **name** `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ **params** `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ **resolver** `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ **pipelineSpec**

Specifying PipelineSpec can be disabled by setting `disable-inline-spec` feature flag.
See Pipeline.spec (API version: tekton.dev/v1)

▼ **status** `string`

Used for cancelling a pipelinerun (and maybe more later on)

▼ taskRunSpecs []object

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

▼ computeResources object

Compute resources to use for this TaskRun

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ 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/> ↗

▼ metadata object

PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask

▼ annotations object

▼ labels object

▼ pipelineTaskName string

▼ podTemplate object

PodTemplate holds pod specific configuration

▼ affinity

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ automountServiceAccountToken boolean

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ dnsConfig `object`

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

▼ nameservers `[]string`

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

▼ options `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name `string`

Name is this DNS resolver option's name. Required.

▼ value `string`

Value is this DNS resolver option's value.

▼ searches `[]string`

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.

▼ dnsPolicy `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Optional: Defaults to true.

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** **required**

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

▼ fieldRef **object**

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`.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ key **string** required

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

▼ hostAliases **[]object**

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

▼ hostnames **[]string**

Hostnames for the above IP address.

▼ ip **string** required

IP address of the host file entry.

▼ **hostNetwork** `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ **imagePullSecrets** `[]object`

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

▼ **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> ↗

▼ **nodeSelector** `object`

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/> ↗

▼ **priorityClassName** `string`

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be

defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName `string`

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName `string`

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings. Optional: Defaults to empty. See type description for default values of each field. See Pod.spec.securityContext (API version: v1)

▼ tolerations `[]object`

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels `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.

▼ matchLabelKeys `[]string`

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).

▼ maxSkew `integer` required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ 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:

- 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` required

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.

▼ whenUnsatisfiable `string` required

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 *more* imbalanced. It's a required field.

▼ volumes

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)

▼ serviceAccountName string

▼ sidecarSpecs []object

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

▼ computeResources object required

The resource requirements to apply to the Sidecar.

▼ claims `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name `string` required

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.

▼ 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/> ↗

▼ **name** **string** required

The name of the Sidecar to override.

▼ **stepSpecs** **[]object**

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

▼ **computeResources** **object** required

The resource requirements to apply to the Step.

▼ **claims** **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ **name** **string** required

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.

▼ 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/> ↗

▼ name **string** **required**

The name of the Step to override.

▼ taskRunTemplate **object**

TaskRunTemplate represent template of taskrun

▼ podTemplate **object**

PodTemplate holds pod specific configuration

▼ affinity

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ automountServiceAccountToken `boolean`

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ dnsConfig `object`

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

▼ nameservers `[]string`

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

▼ options `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name `string`

Name is this DNS resolver option's name. Required.

▼ value `string`

Value is this DNS resolver option's value.

▼ searches `[]string`

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.

▼ dnsPolicy **string**

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks **boolean**

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Optional: Defaults to true.

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secretKeyRef **object**

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

▼ key **string** required

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

▼ hostAliases **[]object**

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

▼ hostnames `[]string`

Hostnames for the above IP address.

▼ ip `string` required

IP address of the host file entry.

▼ hostNetwork `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ imagePullSecrets `[]object`

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

▼ 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> ↗

▼ nodeSelector `object`

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/> ↗

▼ priorityClassName `string`

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName `string`

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName `string`

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings. Optional: Defaults to empty. See type description for default values of each field. See Pod.spec.securityContext (API version: v1)

▼ tolerations `[]object`

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions **[]object**

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

▼ key **string** required

key is the label key that the selector applies to.

▼ operator **string** required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values **[]string**

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.

▼ matchLabels **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.

▼ matchLabelKeys **[]string**

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).

▼ maxSkew **integer** required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ 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:

- 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` required

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.

▼ **whenUnsatisfiable** `string` required

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 *more* imbalanced. It's a required field.

▼ **volumes**

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)

▼ **serviceAccountName** `string`

▼ **timeouts** `object`

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`

▼ finally `string`

Finally sets the maximum allowed duration of this pipeline's finally

▼ pipeline `string`

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.

▼ tasks `string`

Tasks sets the maximum allowed duration of this pipeline's tasks

▼ workspaces `[]object`

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

▼ configMap `object`

ConfigMap represents a configMap that should populate this workspace.

▼ 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 **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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** **required**

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 '..'.

▼ 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

▼ csi **object**

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

▼ driver **string** **required**

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.

▼ fsType **string**

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.

▼ nodePublishSecretRef **object**

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.

▼ 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> ↗

▼ readOnly `boolean`

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes `object`

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

▼ emptyDir `object`

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.

▼ medium `string`

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> ↗

▼ 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> ↗

▼ name string required

Name is the name of the workspace populated by the volume.

▼ persistentVolumeClaim object

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

▼ claimName string required

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> ↗

▼ readOnly boolean

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ projected object

Projected represents a projected volume that should populate this workspace.

▼ defaultMode integer

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.

▼ sources `[]object`

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

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels `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.

▼ 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** **required**

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap **object**

configMap information about the configMap data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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 required

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 '..'.

▼ 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

▼ downwardAPI object

downwardAPI information about the downwardAPI data to project

▼ items []object

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef object

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

▼ apiVersion string

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath string required

Path of the field to select in the specified API version.

▼ 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

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 '..'

▼ resourceFieldRef **object**

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

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secret **object**

secret information about the secret data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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** required

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 '..'.

▼ 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 field specify whether the Secret or its key must be defined

▼ serviceAccountToken `object`

serviceAccountToken is information about the serviceAccountToken data to project

▼ audience `string`

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.

▼ expirationSeconds `integer`

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.

▼ path `string` required

path is the path relative to the mount point of the file to project the token into.

▼ secret **object**

Secret represents a secret that should populate this workspace.

▼ 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 **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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 required

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 '..'.

▼ 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>

▼ 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).

▼ volumeClaimTemplate

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)

▼ status object

PipelineRunStatus defines the observed state of PipelineRun

▼ 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.

▼ childReferences `[]object`

ChildStatusReference is used to point to the statuses of individual TaskRuns and Runs within this PipelineRun.

▼ apiVersion `string`

▼ displayName `string`

DisplayName is a user-facing name of the pipelineTask that may be used to populate a UI.

▼ kind `string`

▼ name `string`

Name is the name of the TaskRun or Run this is referencing.

▼ pipelineTaskName `string`

PipelineTaskName is the name of the PipelineTask this is referencing.

▼ whenExpressions `[]object`

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

▼ 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> ↗

▼ input string

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator string

Operator that represents an Input's relationship to the values

▼ values []string

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

▼ completionTime string

CompletionTime is the time the PipelineRun completed.

▼ conditions []object

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> ↗

▼ **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.

▼ **status** `string` `required`

Status of the condition, one of True, False, Unknown.

▼ **type** `string` `required`

Type of condition.

▼ finallyStartTime `string`

FinallyStartTime is when all non-finally tasks have been completed and only finally tasks are being executed.

▼ observedGeneration `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ pipelineSpec

PipelineSpec contains the exact spec used to instantiate the run. See Pipeline.spec (API version: tekton.dev/v1)

▼ provenance `object`

Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).

▼ featureFlags `object`

FeatureFlags identifies the feature flags that were used during the task/pipeline run

▼ awaitSidecarReadiness `boolean`**▼ coschedule** `string`**▼ disableCredsInit** `boolean`

▼ disableInlineSpec `string`

▼ enableAPIFields `string`

▼ enableArtifacts `boolean`

▼ enableCELInWhenExpression `boolean`

▼ enableConciseResolverSyntax `boolean`

▼ enableKeepPodOnCancel `boolean`

▼ enableKubernetesSidecar `boolean`

▼ enableParamEnum `boolean`

▼ enableProvenanceInStatus `boolean`

▼ enableStepActions `boolean`

EnableStepActions is a no-op flag since StepActions are stable

▼ enforceNonfalsifiability `string`

▼ **maxResultSize** `integer`

▼ **requireGitSSHSecretKnownHosts** `boolean`

▼ **resultExtractionMethod** `string`

▼ **runningInEnvWithInjectedSidecars** `boolean`

▼ **sendCloudEventsForRuns** `boolean`

▼ **setSecurityContext** `boolean`

▼ **setSecurityContextReadOnlyRootFilesystem** `boolean`

▼ **verificationNoMatchPolicy** `string`

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

▼ **refSource** `object`

RefSource identifies the source where a remote task/pipeline came from.

▼ **digest** **object**

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1":

"f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ **entryPoint** **string**

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"

▼ **uri** **string**

URI indicates the identity of the source of the build definition. Example:

"https://github.com/tektoncd/catalog ↗"

▼ **results** **[]object**

PipelineRunResult used to describe the results of a pipeline

▼ **name** **string** **required**

Name is the result's name as declared by the Pipeline

▼ **value** **required**

Value is the result returned from the execution of this PipelineRun

▼ skippedTasks `[]object`

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.

▼ name `string` required

Name is the Pipeline Task name

▼ reason `string` required

Reason is the cause of the PipelineTask being skipped.

▼ whenExpressions `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ **values** `[]string`

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

▼ **spanContext** `object`

SpanContext contains tracing span context fields

▼ **startTime** `string`

StartTime is the time the PipelineRun is actually started.

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.

Deprecated: Please use `v1.PipelineRun` instead.

`v1beta1`

`version`

▼ **spec** `object`

PipelineRunSpec defines the desired state of PipelineRun

▼ **params** `[]object`

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

▼ **name** `string` `required`

▼ **value** required

▼ **pipelineRef** object

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

▼ **apiVersion** string

API version of the referent

▼ **bundle** string

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.

▼ **name** string

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ **params** []object

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

▼ **name** string required

▼ **value** required

▼ resolver `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ pipelineSpec

Specifying PipelineSpec can be disabled by setting `disable-inline-spec` feature flag. See Pipeline.spec (API version: tekton.dev/v1beta1)

▼ podTemplate `object`

PodTemplate holds pod specific configuration

▼ affinity

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ automountServiceAccountToken `boolean`

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ dnsConfig `object`

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

▼ nameservers `[]string`

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

▼ options `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name `string`

Name is this DNS resolver option's name. Required.

▼ value `string`

Value is this DNS resolver option's value.

▼ searches `[]string`

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.

▼ dnsPolicy `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links.

Optional: Defaults to true.

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef `object`

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.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ hostAliases `[]object`

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

▼ hostnames `[]string`

Hostnames for the above IP address.

▼ ip `string` required

IP address of the host file entry.

▼ hostNetwork `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ imagePullSecrets `[]object`

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

▼ 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> ↗

▼ nodeSelector object

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/> ↗

▼ priorityClassName string

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName string

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName string

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings.

Optional: Defaults to empty. See type description for default values of each field.

See Pod.spec.securityContext (API version: v1)

▼ tolerations []object

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values.
Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels 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.

▼ matchLabelKeys []string

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).

▼ maxSkew integer required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ 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:

- 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` required

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.

▼ whenUnsatisfiable `string` required

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 *more* imbalanced. It's a required field.

▼ volumes

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)

▼ resources `[]object`

PipelineResourceBinding connects a reference to an instance of a PipelineResource with a PipelineResource dependency that the Pipeline has declared

Deprecated: Unused, preserved only for backwards compatibility

▼ name `string`

Name is the name of the PipelineResource in the Pipeline's declaration

▼ resourceRef object

ResourceRef is a reference to the instance of the actual PipelineResource that should be used

▼ apiVersion string

API version of the referent

▼ name string

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ resourceSpec object

ResourceSpec is specification of a resource that should be created and consumed by the task

▼ description string

Description is a user-facing description of the resource that may be used to populate a UI.

▼ params []object required

ResourceParam declares a string value to use for the parameter called Name, and is used in the specific context of PipelineResources.

Deprecated: Unused, preserved only for backwards compatibility

▼ **name** `string` required

▼ **value** `string` required

▼ **secrets** `[]object`

SecretParam indicates which secret can be used to populate a field of the resource

Deprecated: Unused, preserved only for backwards compatibility

▼ **fieldName** `string` required

▼ **secretKey** `string` required

▼ **secretName** `string` required

▼ **type** `string` required

PipelineResourceType represents the type of endpoint the pipelineResource is, so that the controller will know this pipelineResource shouldx be fetched and optionally what additional metatdata should be provided for it.

Deprecated: Unused, preserved only for backwards compatibility

▼ **serviceAccountName** `string`

▼ status **string**

Used for cancelling a pipelinerun (and maybe more later on)

▼ taskRunSpecs **[]object**

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

▼ computeResources **object**

Compute resources to use for this TaskRun

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** required

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.

▼ limits **object**

Limits describes the maximum amount of compute resources allowed.

More info: [https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/ ↗](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/> ↗

▼ metadata **object**

PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask

▼ annotations **object****▼ labels** **object****▼ pipelineTaskName** **string****▼ sidecarOverrides** **[]object**

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

▼ name **string** **required**

The name of the Sidecar to override.

▼ resources **object** **required**

The resource requirements to apply to the Sidecar.

▼ claims `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name `string` required

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.

▼ 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/> ↗

▼ stepOverrides `[]object`

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

▼ name `string` required

The name of the Step to override.

▼ resources `object` required

The resource requirements to apply to the Step.

▼ claims `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name `string` required

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.

▼ 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/> ↗

▼ taskPodTemplate **object**

PodTemplate holds pod specific configuration

▼ affinity

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ automountServiceAccountToken **boolean**

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ dnsConfig **object**

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

▼ **nameservers** `[]string`

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

▼ **options** `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ **name** `string`

Name is this DNS resolver option's name. Required.

▼ **value** `string`

Value is this DNS resolver option's value.

▼ **searches** `[]string`

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.

▼ **dnsPolicy** `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Optional: Defaults to true.

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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`.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ key **string** required

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

▼ hostAliases **[]object**

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

▼ hostnames **[]string**

Hostnames for the above IP address.

▼ ip **string** required

IP address of the host file entry.

▼ **hostNetwork** `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ **imagePullSecrets** `[]object`

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

▼ **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> ↗

▼ **nodeSelector** `object`

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/> ↗

▼ **priorityClassName** `string`

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be

defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName `string`

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName `string`

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings. Optional: Defaults to empty. See type description for default values of each field. See Pod.spec.securityContext (API version: v1)

▼ tolerations `[]object`

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels `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.

▼ matchLabelKeys `[]string`

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).

▼ maxSkew `integer` required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ 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:

- 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` required

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.

▼ whenUnsatisfiable `string` required

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 *more* imbalanced. It's a required field.

▼ volumes

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)

▼ taskServiceAccountName string

▼ timeout string

Timeout is the Time after which the Pipeline times out. Defaults to never. Refer to Go's ParseDuration documentation for expected format:

<https://golang.org/pkg/time/#ParseDuration> ↗

Deprecated: use pipelineRunSpec.Timeouts.Pipeline instead

▼ timeouts **object**

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

▼ finally **string**

Finally sets the maximum allowed duration of this pipeline's finally

▼ pipeline **string**

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.

▼ tasks **string**

Tasks sets the maximum allowed duration of this pipeline's tasks

▼ workspaces **[]object**

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

▼ configMap **object**

ConfigMap represents a configMap that should populate this workspace.

▼ 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 `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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` required

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 `'..'`.

▼ 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

▼ **csi** **object**

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

▼ **driver** **string** **required**

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.

▼ **fsType** **string**

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.

▼ **nodePublishSecretRef** **object**

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.

▼ **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> ↗

▼ readOnly **boolean**

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes **object**

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

▼ emptyDir **object**

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.

▼ medium **string**

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> ↗

▼ 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> ↗

▼ **name** **string** required

Name is the name of the workspace populated by the volume.

▼ **persistentVolumeClaim** **object**

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

▼ **claimName** **string** required

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> ↗

▼ **readOnly** **boolean**

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ **projected** **object**

Projected represents a projected volume that should populate this workspace.

▼ **defaultMode** **integer**

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.

▼ sources `[]object`

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

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions `[]object`

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

▼ **key** `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]string`

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.

▼ **matchLabels** `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.

▼ 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` required

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap `object`

configMap information about the configMap data to project

▼ items `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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** required

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 '..'.

▼ 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

▼ downwardAPI **object**

downwardAPI information about the downwardAPI data to project

▼ items **[]object**

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef **object**

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** required

Path of the field to select in the specified API version.

▼ 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

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 '..'

▼ **resourceFieldRef** `object`

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

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ secret **object**

secret information about the secret data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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** required

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 '..'.

▼ 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 field specify whether the Secret or its key must be defined

▼ serviceAccountToken **object**

serviceAccountToken is information about the serviceAccountToken data to project

▼ audience **string**

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.

▼ expirationSeconds **integer**

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.

▼ **path** **string** **required**

path is the path relative to the mount point of the file to project the token into.

▼ **secret** **object**

Secret represents a secret that should populate this workspace.

▼ **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** **[]object**

Maps a string key to a path within a volume.

▼ **key** **string** **required**

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` required

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 '..'.

▼ **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> ↗

▼ **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).

▼ **volumeClaimTemplate**

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)

▼ status **object**

PipelineRunStatus defines the observed state of PipelineRun

▼ 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.

▼ childReferences **[]object**

ChildStatusReference is used to point to the statuses of individual TaskRuns and Runs within this PipelineRun.

▼ apiVersion **string**

▼ displayName **string**

DisplayName is a user-facing name of the pipelineTask that may be used to populate a UI.

▼ kind **string**

▼ name **string**

Name is the name of the TaskRun or Run this is referencing.

▼ pipelineTaskName `string`

PipelineTaskName is the name of the PipelineTask this is referencing.

▼ whenExpressions `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ completionTime `string`

CompletionTime is the time the PipelineRun completed.

▼ conditions `[]object`

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> ↗

▼ 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.

▼ status `string` `required`

Status of the condition, one of True, False, Unknown.

▼ type `string` required

Type of condition.

▼ finallyStartTime `string`

FinallyStartTime is when all non-finally tasks have been completed and only finally tasks are being executed.

▼ observedGeneration `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ pipelineResults `[]object`

PipelineRunResult used to describe the results of a pipeline

▼ name `string` required

Name is the result's name as declared by the Pipeline

▼ value required

Value is the result returned from the execution of this PipelineRun

▼ pipelineSpec

PipelineSpec contains the exact spec used to instantiate the run. See Pipeline.spec (API version: tekton.dev/v1beta1)

▼ provenance **object**

Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).

▼ configSource **object**

Deprecated: Use RefSource instead

▼ digest **object**

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1":

"f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint **string**

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"

▼ uri **string**

URI indicates the identity of the source of the build definition. Example:

"https://github.com/tektoncd/catalog ↗"

▼ featureFlags **object**

FeatureFlags identifies the feature flags that were used during the task/pipeline run

▼ awaitSidecarReadiness **boolean**

▼ **coschedule** `string`

▼ **disableCredsInit** `boolean`

▼ **disableInlineSpec** `string`

▼ **enableAPIFields** `string`

▼ **enableArtifacts** `boolean`

▼ **enableCELInWhenExpression** `boolean`

▼ **enableConciseResolverSyntax** `boolean`

▼ **enableKeepPodOnCancel** `boolean`

▼ **enableKubernetesSidecar** `boolean`

▼ **enableParamEnum** `boolean`

▼ **enableProvenanceInStatus** `boolean`

▼ enableStepActions `boolean`

EnableStepActions is a no-op flag since StepActions are stable

▼ enforceNonfalsifiability `string`**▼ maxResultSize** `integer`**▼ requireGitSSHSecretKnownHosts** `boolean`**▼ resultExtractionMethod** `string`**▼ runningInEnvWithInjectedSidecars** `boolean`**▼ sendCloudEventsForRuns** `boolean`**▼ setSecurityContext** `boolean`**▼ setSecurityContextReadOnlyRootFilesystem** `boolean`**▼ verificationNoMatchPolicy** `string`

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

▼ refSource object

RefSource identifies the source where a remote task/pipeline came from.

▼ digest object

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1":

"f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint string

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"

▼ uri string

URI indicates the identity of the source of the build definition. Example:

"<https://github.com/tektoncd/catalog> ↗"

▼ runs object

Runs is a map of PipelineRunRunStatus with the run name as the key

Deprecated: use ChildReferences instead. As of v0.45.0, this field is no longer populated and is only included for backwards compatibility with older server versions.

▼ skippedTasks `[]object`

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.

▼ name `string` required

Name is the Pipeline Task name

▼ reason `string` required

Reason is the cause of the PipelineTask being skipped.

▼ whenExpressions `[]object`

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

▼ 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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ **values** `[]string`

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

▼ **spanContext** `object`

SpanContext contains tracing span context fields

▼ **startTime** `string`

StartTime is the time the PipelineRun is actually started.

▼ **taskRuns** `object`

TaskRuns is a map of PipelineRunTaskRunStatus with the taskRun name as the key.

Deprecated: use ChildReferences instead. As of v0.45.0, this field is no longer populated and is only included for backwards compatibility with older server versions.



TaskRun [tekton.dev/v1]

tekton.dev group

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.

v1 version

▼ spec object

TaskRunSpec defines the desired state of TaskRun

▼ computeResources object

Compute resources to use for this TaskRun

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ 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/> ↗

▼ debug object

TaskRunDebug defines the breakpoint config for a particular TaskRun

▼ breakpoints object

TaskBreakpoints defines the breakpoint config for a particular Task

▼ beforeSteps []string

▼ onFailure string

if enabled, pause TaskRun on failure of a step failed step will not exit

▼ params []object

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

▼ name string required

▼ value required

▼ podTemplate object

PodTemplate holds pod specific configuration

▼ affinity

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ automountServiceAccountToken boolean

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ dnsConfig object

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

▼ nameservers []string

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

▼ options `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name `string`

Name is this DNS resolver option's name. Required.

▼ value `string`

Value is this DNS resolver option's value.

▼ searches `[]string`

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.

▼ dnsPolicy `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links.

Optional: Defaults to true.

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef `object`

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.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ `hostAliases` `[]object`

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

▼ `hostnames` `[]string`

Hostnames for the above IP address.

▼ `ip` `string` `required`

IP address of the host file entry.

▼ `hostNetwork` `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ `imagePullSecrets` `[]object`

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

▼ `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> ↗

▼ nodeSelector object

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/> ↗

▼ priorityClassName string

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName string

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName string

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings.

Optional: Defaults to empty. See type description for default values of each field.

See Pod.spec.securityContext (API version: v1)

▼ tolerations []object

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values.
Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels 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.

▼ matchLabelKeys []string

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).

▼ maxSkew integer required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ 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:

- 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` required

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.

▼ whenUnsatisfiable string required

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 *more* imbalanced. It's a required field.

▼ volumes

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)

▼ retries integer

Retries represents how many times this TaskRun should be retried in the event of task failure.

▼ serviceAccountName string

▼ sidecarSpecs []object

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

▼ computeResources object required

The resource requirements to apply to the Sidecar.

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ 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/> ↗

▼ **name** **string** required

The name of the Sidecar to override.

▼ **status** **string**

Used for cancelling a TaskRun (and maybe more later on)

▼ **statusMessage** **string**

Status message for cancellation.

▼ **stepSpecs** **[]object**

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

▼ **computeResources** **object** required

The resource requirements to apply to the Step.

▼ **claims** **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ **name** **string** required

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.

▼ 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/> ↗

▼ name string required

The name of the Step to override.

▼ taskRef object

no more than one of the TaskRef and TaskSpec may be specified.

▼ 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:

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

▼ name string

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params []object

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

▼ name string required**▼ value** required**▼ resolver** string

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ taskSpec

Specifying TaskSpec can be disabled by setting `disable-inline-spec` feature flag. See Task.spec (API version: tekton.dev/v1)

▼ timeout `string`

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> ↗

▼ workspaces `[]object`

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

▼ configMap `object`

ConfigMap represents a configMap that should populate this workspace.

▼ 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 `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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** required

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 '..'.

▼ 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

▼ csi **object**

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

▼ **driver** `string` required

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.

▼ **fsType** `string`

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.

▼ **nodePublishSecretRef** `object`

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.

▼ **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> ↗

▼ **readOnly** `boolean`

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes object

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

▼ emptyDir object

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.

▼ medium string

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> ↗

▼ 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> ↗

▼ name string required

Name is the name of the workspace populated by the volume.

▼ persistentVolumeClaim **object**

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

▼ claimName **string** required

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> ↗

▼ readOnly **boolean**

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ projected **object**

Projected represents a projected volume that should populate this workspace.

▼ defaultMode **integer**

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.

▼ sources `[]object`

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

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]string`

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.

▼ **matchLabels** `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.

▼ **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** **required**

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap **object**

configMap information about the configMap data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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` required

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 '..'.

▼ **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

▼ **downwardAPI** `object`

downwardAPI information about the downwardAPI data to project

▼ items []object

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef object

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

▼ apiVersion string

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath string required

Path of the field to select in the specified API version.

▼ 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

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 '..'

▼ resourceFieldRef **object**

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

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secret **object**

secret information about the secret data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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** **required**

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 '..'.

▼ 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 field specify whether the Secret or its key must be defined

▼ serviceAccountToken `object`

serviceAccountToken is information about the serviceAccountToken data to project

▼ audience `string`

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.

▼ expirationSeconds `integer`

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.

▼ path `string` required

path is the path relative to the mount point of the file to project the token into.

▼ secret **object**

Secret represents a secret that should populate this workspace.

▼ 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 **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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 required

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 '..'.

▼ 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>

▼ 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).

▼ volumeClaimTemplate

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)

▼ status object

TaskRunStatus defines the observed state of TaskRun

▼ 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.

▼ artifacts `object`

Artifacts are the list of artifacts written out by the task's containers

▼ inputs `[]object`

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

▼ buildOutput `boolean`

Indicate if the artifact is a build output or a by-product

▼ name `string`

The artifact's identifying category name

▼ values `[]object`

ArtifactValue represents a specific value or data element within an Artifact.

▼ digest `object`

▼ uri `string`

▼ outputs `[]object`

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

▼ buildOutput `boolean`

Indicate if the artifact is a build output or a by-product

▼ name `string`

The artifact's identifying category name

▼ values `[]object`

ArtifactValue represents a specific value or data element within an Artifact.

▼ digest `object`**▼ uri** `string`**▼ completionTime** `string`

CompletionTime is the time the build completed.

▼ conditions `[]object`

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> ↗

▼ **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.

▼ **status** `string` `required`

Status of the condition, one of True, False, Unknown.

▼ **type** `string` `required`

Type of condition.

▼ observedGeneration integer

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ podName string required

PodName is the name of the pod responsible for executing this task's steps.

▼ provenance object

Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).

▼ featureFlags object

FeatureFlags identifies the feature flags that were used during the task/pipeline run

▼ awaitSidecarReadiness boolean**▼ coschedule** string**▼ disableCredsInit** boolean**▼ disableInlineSpec** string**▼ enableAPIFields** string

▼ **enableArtifacts** **boolean**

▼ **enableCELInWhenExpression** **boolean**

▼ **enableConciseResolverSyntax** **boolean**

▼ **enableKeepPodOnCancel** **boolean**

▼ **enableKubernetesSidecar** **boolean**

▼ **enableParamEnum** **boolean**

▼ **enableProvenanceInStatus** **boolean**

▼ **enableStepActions** **boolean**

EnableStepActions is a no-op flag since StepActions are stable

▼ **enforceNonfalsifiability** **string**

▼ **maxResultSize** **integer**

▼ **requireGitSSHSecretKnownHosts** **boolean**

▼ **resultExtractionMethod** `string`

▼ **runningInEnvWithInjectedSidecars** `boolean`

▼ **sendCloudEventsForRuns** `boolean`

▼ **setSecurityContext** `boolean`

▼ **setSecurityContextReadOnlyRootFilesystem** `boolean`

▼ **verificationNoMatchPolicy** `string`

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

▼ **refSource** `object`

RefSource identifies the source where a remote task/pipeline came from.

▼ **digest** `object`

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ **entryPoint** string

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"

▼ **uri** string

URI indicates the identity of the source of the build definition. Example: "<https://github.com/tektoncd/catalog> ↗"

▼ **results** []object

TaskRunResult used to describe the results of a task

▼ **name** string required

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** required

Value the given value of the result

▼ retriesStatus

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.

▼ sidecars []object

SidecarState reports the results of running a sidecar in a Task.

▼ container string

▼ imageID string

▼ name string

▼ running object

Details about a running container

▼ startedAt string

Time at which the container was last (re-)started

▼ terminated object

Details about a terminated container

▼ containerID string

Container's ID in the format '://<container_id>'

▼ **exitCode** **integer** required

Exit status from the last termination of the container

▼ **finishedAt** **string**

Time at which the container last terminated

▼ **message** **string**

Message regarding the last termination of the container

▼ **reason** **string**

(brief) reason from the last termination of the container

▼ **signal** **integer**

Signal from the last termination of the container

▼ **startedAt** **string**

Time at which previous execution of the container started

▼ **waiting** **object**

Details about a waiting container

▼ **message** **string**

Message regarding why the container is not yet running.

▼ **reason** `string`

(brief) reason the container is not yet running.

▼ **spanContext** `object`

SpanContext contains tracing span context fields

▼ **startTime** `string`

StartTime is the time the build is actually started.

▼ **steps** `[]object`

StepState reports the results of running a step in a Task.

▼ **container** `string`

▼ **imageID** `string`

▼ **inputs** `[]object`

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

▼ **buildOutput** `boolean`

Indicate if the artifact is a build output or a by-product

▼ name `string`

The artifact's identifying category name

▼ values `[]object`

ArtifactValue represents a specific value or data element within an Artifact.

▼ digest `object`**▼ uri** `string`**▼ name** `string`**▼ outputs** `[]object`

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

▼ buildOutput `boolean`

Indicate if the artifact is a build output or a by-product

▼ name `string`

The artifact's identifying category name

▼ values `[]object`

ArtifactValue represents a specific value or data element within an Artifact.

▼ digest **object**

▼ uri **string**

▼ provenance **object**

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.

▼ featureFlags **object**

FeatureFlags identifies the feature flags that were used during the task/pipeline run

▼ awaitSidecarReadiness **boolean**

▼ coschedule **string**

▼ disableCredsInit **boolean**

▼ disableInlineSpec **string**

▼ enableAPIFields **string**

▼ **enableArtifacts** `boolean`

▼ **enableCELInWhenExpression** `boolean`

▼ **enableConciseResolverSyntax** `boolean`

▼ **enableKeepPodOnCancel** `boolean`

▼ **enableKubernetesSidecar** `boolean`

▼ **enableParamEnum** `boolean`

▼ **enableProvenanceInStatus** `boolean`

▼ **enableStepActions** `boolean`

EnableStepActions is a no-op flag since StepActions are stable

▼ **enforceNonfalsifiability** `string`

▼ **maxResultSize** `integer`

▼ **requireGitSSHSecretKnownHosts** **boolean**

▼ **resultExtractionMethod** **string**

▼ **runningInEnvWithInjectedSidecars** **boolean**

▼ **sendCloudEventsForRuns** **boolean**

▼ **setSecurityContext** **boolean**

▼ **setSecurityContextReadOnlyRootFilesystem**
boolean

▼ **verificationNoMatchPolicy** **string**

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

▼ **refSource** **object**

RefSource identifies the source where a remote task/pipeline came from.

▼ digest **object**

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint **string**

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"

▼ uri **string**

URI indicates the identity of the source of the build definition. Example: "<https://github.com/tektoncd/catalog>"

▼ results **[]object**

TaskRunResult used to describe the results of a task

▼ name **string** **required**

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 **required**

Value the given value of the result

▼ **running** **object**

Details about a running container

▼ **startedAt** **string**

Time at which the container was last (re-)started

▼ **terminated** **object**

Details about a terminated container

▼ **containerID** **string**

Container's ID in the format '://<container_id>'

▼ **exitCode** **integer** **required**

Exit status from the last termination of the container

▼ **finishedAt** **string**

Time at which the container last terminated

▼ **message** **string**

Message regarding the last termination of the container

▼ **reason** **string**

(brief) reason from the last termination of the container

▼ **signal** `integer`

Signal from the last termination of the container

▼ **startedAt** `string`

Time at which previous execution of the container started

▼ **terminationReason** `string`

▼ **waiting** `object`

Details about a waiting container

▼ **message** `string`

Message regarding why the container is not yet running.

▼ **reason** `string`

(brief) reason the container is not yet running.

▼ **taskSpec** `object`

TaskSpec contains the Spec from the dereferenced Task definition used to instantiate this TaskRun.

▼ **description** `string`

Description is a user-facing description of the task that may be used to populate a UI.

▼ displayName `string`

DisplayName is a user-facing name of the task that may be used to populate a UI.

▼ params `[]object`

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

▼ default

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ properties **object**

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

▼ type **string**

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ results **[]object**

TaskResult used to describe the results of a task

▼ description **string**

Description is a human-readable description of the result

▼ name **string** **required**

Name the given name

▼ properties **object**

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

▼ 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 expression used to retrieve the value of the result from an underlying Step.

▼ sidecars `[]object`

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

▼ args `[]string`

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 `[]string`

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> ↗

▼ computeResources **object**

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** required

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.

▼ 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/> ↗

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

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.

▼ configMapKeyRef `object`

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** required

Path of the field to select in the specified API version.

▼ resourceFieldRef 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.

▼ containerName string

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource string required

Required: resource to select

▼ secretKeyRef object

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

▼ key string required

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

▼ envFrom `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef `object`

The ConfigMap to select from

▼ 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 must be defined

▼ **prefix** `string`

An optional identifier to prepend to each key in the ConfigMap.
Must be a C_IDENTIFIER.

▼ **secretRef** `object`

The Secret to select from

▼ **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

▼ **image** `string`

Image reference name. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ **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#updating-images> ↗

▼ lifecycle **object**

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

▼ 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> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP.

You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **sleep** `object`

Sleep represents a duration that the container should sleep.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ 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> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP.

You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **sleep** `object`

Sleep represents a duration that the container should sleep.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ livenessProbe **object**

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> ↗

▼ exec **object**

Exec specifies a command to execute in the container.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

gRPC specifies a gRPC HealthCheckRequest.

▼ port integer required

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.

▼ httpGet object

HTTPGet specifies an HTTP GET request to perform.

▼ host string

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders []object

HTTPHeader describes a custom header to be used in HTTP probes

▼ name string required

The header field name. This will be canonicalized upon output, so case-variant names will be

understood as the same header.

▼ **value** **string** required

The header field value

▼ **path** **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** **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> ↗

▼ **periodSeconds** **integer**

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ successThreshold integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ tcpSocket object

TCPSocket specifies a connection to a TCP port.

▼ host string

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ terminationGracePeriodSeconds integer

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate.

Minimum value is 1. `spec.terminationGracePeriodSeconds` is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **name** `string` required

Name of the Sidecar specified as a DNS_LABEL. Each Sidecar in a Task must have a unique name (DNS_LABEL). Cannot be updated.

▼ **ports** `[]object`

ContainerPort represents a network port in a single container.

▼ **containerPort** `integer` required

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.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ 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/pod-lifecycle#container-probes> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ **port** **integer** **required**

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.

▼ **httpGet** **object**

HTTPGet specifies an HTTP GET request to perform.

▼ **host** **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

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

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** integer

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** object

TCPSocket specifies a connection to a TCP port.

▼ **host** string

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ **terminationGracePeriodSeconds** integer

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ restartPolicy string

RestartPolicy refers to kubernetes RestartPolicy. It can only be set for an initContainer and must have it's policy set to "Always". It is currently left optional to help support Kubernetes versions prior to 1.29 when this feature was introduced.

▼ script string

Script is the contents of an executable file to execute.

If Script is not empty, the Step cannot have an Command or Args.

▼ securityContext object

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/> ↗

▼ allowPrivilegeEscalation **boolean**

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs 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.

▼ appArmorProfile **object**

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.

▼ localhostProfile **string**

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".

▼ type **string** required

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.

▼ 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.

▼ add `[]string`

Added capabilities

▼ drop `[]string`

Removed capabilities

▼ 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 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 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.

▼ **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.

▼ **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.

▼ **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` required

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.

▼ **windowsOptions** `object`

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.

▼ **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.

▼ **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 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.

▼ startupProbe `object`

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> ↗

▼ exec `object`

Exec specifies a command to execute in the container.

▼ command `[]string`

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies a GRPC HealthCheckRequest.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies an HTTP GET request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

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

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ initialDelaySeconds `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> ↗

▼ periodSeconds `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ successThreshold `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ tcpSocket `object`

TCPSocket specifies a connection to a TCP port.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port `required`

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.

▼ **terminationGracePeriodSeconds** **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **stdin** **boolean**

Whether this Sidecar should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the Sidecar will always result in EOF. Default is false.

▼ stdinOnce `boolean`

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on Sidecar start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the Sidecar is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath `string`

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, such as an assertion failure message. Will be truncated by the node 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.

▼ terminationMessagePolicy `string`

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 Sidecar log output if the termination message file is empty and the Sidecar exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to `File`. Cannot be updated.

▼ tty `boolean`

Whether this Sidecar should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ volumeDevices []object

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

▼ devicePath string required

devicePath is the path inside of the container that the device will be mapped to.

▼ name string required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []object

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath string required

Path within the container at which the volume should be mounted.
Must not contain ':'.

▼ mountPropagation string

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` required

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.

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.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ workingDir `string`

Sidecar'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.

▼ workspaces `[]object`

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

▼ mountPath `string` required

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.

▼ name `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ stepTemplate `object`

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

▼ args `[]string`

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> ↗

▼ command `[]string`

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> ↗

▼ computeResources `object`

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

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ 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/> ↗

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** *required*

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** *required*

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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** required

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ `envFrom` `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ `configMapRef` `object`

The ConfigMap to select from

▼ `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 must be defined

▼ prefix `string`

An optional identifier to prepend to each key in the ConfigMap.
Must be a C_IDENTIFIER.

▼ secretRef `object`

The Secret to select from

▼ 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

▼ image `string`

Image reference name. More info:
<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ 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#updating-images> ↗

▼ 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/> ↗

▼ allowPrivilegeEscalation **boolean**

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs 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.

▼ appArmorProfile **object**

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.

▼ localhostProfile **string**

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".

▼ type **string** required

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.

▼ 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.

▼ add **[]string**

Added capabilities

▼ drop **[]string**

Removed capabilities

▼ 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 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 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.

▼ **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.

▼ **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`.

▼ **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` required

`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.

▼ **windowsOptions** `object`

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`.

▼ **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.

▼ 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 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.

▼ volumeDevices `[]object`

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

▼ devicePath `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ name **string** **required**

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts **[]object**

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath **string** **required**

Path within the container at which the volume should be mounted.

Must not contain ':'.

▼ mountPropagation **string**

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** **required**

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.

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`.

▼ `subPath` `string`

Path within the volume from which the container's volume should be mounted. Defaults to `""` (volume's root).

▼ `subPathExpr` `string`

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.

▼ `workingDir` `string`

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.

▼ steps []object

Step runs a subcomponent of a Task

▼ args []string

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> ↗

▼ command []string

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> ↗

▼ computeResources object

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

▼ claims []object

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name string required

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.

▼ 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/> ↗

▼ env object

EnvVar represents an environment variable present in a Container.

▼ name string required

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.

▼ configMapKeyRef object

Selects a key of a ConfigMap.

▼ key string required

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

▼ fieldRef `object`

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.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** **required**

Required: resource to select

▼ secretKeyRef **object**

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

▼ key **string** **required**

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

▼ `envFrom` `[]object`

EnvFromSource represents the source of a set of ConfigMaps

▼ `configMapRef` `object`

The ConfigMap to select from

▼ `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 must be defined

▼ **prefix** `string`

An optional identifier to prepend to each key in the ConfigMap.
Must be a C_IDENTIFIER.

▼ **secretRef** `object`

The Secret to select from

▼ **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

▼ **image** `string`

Docker image name. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ **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#updating-images> ↗

▼ **name** **string** required

Name of the Step specified as a DNS_LABEL. Each Step in a Task must have a unique name.

▼ **onError** **string**

OnError defines the exiting behavior of a container on error can be set to [continue | stopAndFail]

▼ **params** **[]object**

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

▼ **name** **string** required

▼ **value** required

▼ **ref** **object**

Contains the reference to an existing StepAction.

▼ **name** **string**

Name of the referenced step

▼ params **[]object**

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

▼ name **string** **required****▼ value** **required****▼ resolver** **string**

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ results **[]object**

StepResult used to describe the Results of a Step.

▼ description **string**

Description is a human-readable description of the result

▼ name **string** **required**

Name the given name

▼ properties **object**

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

▼ type string

The possible types are 'string', 'array', and 'object', with 'string' as the default.

▼ 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/>

▼ allowPrivilegeEscalation boolean

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs 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.

▼ appArmorProfile object

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.

▼ localhostProfile `string`

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".

▼ type `string` `required`

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.

▼ 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.

▼ add `[]string`

Added capabilities

▼ drop `[]string`

Removed capabilities

▼ 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 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 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.

▼ 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.

▼ **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.

▼ **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** **required**

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.

▼ **windowsOptions** **object**

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.

▼ 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.

▼ 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 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.

▼ stderrConfig `object`

Stores configuration for the stderr stream of the step.

▼ path string

Path to duplicate stdout stream to on container's local filesystem.

▼ stdoutConfig object

Stores configuration for the stdout stream of the step.

▼ path string

Path to duplicate stdout stream to on container's local filesystem.

▼ timeout string

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> ↗

▼ volumeDevices []object

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

▼ devicePath string required

devicePath is the path inside of the container that the device will be mapped to.

▼ name string required

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts []object

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath string required

Path within the container at which the volume should be mounted.

Must not contain ':'.

▼ mountPropagation string

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 required

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.

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`.

▼ **subPath** `string`

Path within the volume from which the container's volume should be mounted. Defaults to `""` (volume's root).

▼ **subPathExpr** `string`

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.

▼ **when** `[]object`

`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

▼ **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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workingDir `string`

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.

▼ workspaces `[]object`

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

▼ mountPath `string` required

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.

▼ **name** `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ **volumes**

Volumes is a collection of volumes that are available to mount into the steps of the build. See Pod.spec.volumes (API version: v1)

▼ **workspaces** `[]object`

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

▼ **description** `string`

Description is an optional human readable description of this volume.

▼ **mountPath** `string`

MountPath overrides the directory that the volume will be made available at.

▼ **name** `string` required

Name is the name by which you can bind the volume at runtime.

▼ optional `boolean`

Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.

▼ readOnly `boolean`

ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.

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.

Deprecated: Please use `v1.TaskRun` instead.

`v1beta1``version`**▼ spec** `object`

TaskRunSpec defines the desired state of TaskRun

▼ computeResources `object`

Compute resources to use for this TaskRun

▼ claims `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name `string` `required`

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.

▼ 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/> ↗

▼ debug **object**

TaskRunDebug defines the breakpoint config for a particular TaskRun

▼ breakpoints **object**

TaskBreakpoints defines the breakpoint config for a particular Task

▼ beforeSteps **[]string****▼ onFailure** **string**

if enabled, pause TaskRun on failure of a step failed step will not exit

▼ **params** []object

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

▼ **name** string required

▼ **value** required

▼ **podTemplate** object

PodTemplate holds pod specific configuration

▼ **affinity**

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ **automountServiceAccountToken** boolean

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ **dnsConfig** object

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

▼ **nameservers** []string

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

▼ options `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name `string`

Name is this DNS resolver option's name. Required.

▼ value `string`

Value is this DNS resolver option's value.

▼ searches `[]string`

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.

▼ dnsPolicy `string`

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks `boolean`

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links.

Optional: Defaults to true.

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef `object`

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.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ `hostAliases` `[]object`

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

▼ `hostnames` `[]string`

Hostnames for the above IP address.

▼ `ip` `string` required

IP address of the host file entry.

▼ `hostNetwork` `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ `imagePullSecrets` `[]object`

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

▼ `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> ↗

▼ nodeSelector object

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/> ↗

▼ priorityClassName string

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName string

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName string

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings.

Optional: Defaults to empty. See type description for default values of each field.

See Pod.spec.securityContext (API version: v1)

▼ tolerations []object

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values.
Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels **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.

▼ matchLabelKeys **[]string**

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).

▼ maxSkew **integer** required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ 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:

- 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` required

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.

▼ whenUnsatisfiable string required

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 *more* imbalanced. It's a required field.

▼ volumes

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)

▼ resources object

Deprecated: Unused, preserved only for backwards compatibility

▼ inputs []object

TaskResourceBinding points to the PipelineResource that will be used for the Task input or output called Name.

Deprecated: Unused, preserved only for backwards compatibility

▼ name `string`

Name is the name of the PipelineResource in the Pipeline's declaration

▼ paths `[]string`

Paths will probably be removed in #1284, and then PipelineResourceBinding can be used instead. The optional Path field corresponds to a path on disk at which the Resource can be found (used when providing the resource via mounted volume, overriding the default logic to fetch the Resource).

▼ resourceRef `object`

ResourceRef is a reference to the instance of the actual PipelineResource that should be used

▼ apiVersion `string`

API version of the referent

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ resourceSpec `object`

ResourceSpec is specification of a resource that should be created and consumed by the task

▼ description `string`

Description is a user-facing description of the resource that may be used to populate a UI.

▼ **params** `[]object` required

ResourceParam declares a string value to use for the parameter called Name, and is used in the specific context of PipelineResources.

Deprecated: Unused, preserved only for backwards compatibility

▼ **name** `string` required

▼ **value** `string` required

▼ **secrets** `[]object`

SecretParam indicates which secret can be used to populate a field of the resource

Deprecated: Unused, preserved only for backwards compatibility

▼ **fieldName** `string` required

▼ **secretKey** `string` required

▼ **secretName** `string` required

▼ type `string` required

PipelineResourceType represents the type of endpoint the pipelineResource is, so that the controller will know this pipelineResource shouldx be fetched and optionally what additional metatdata should be provided for it.

Deprecated: Unused, preserved only for backwards compatibility

▼ outputs `[]object`

TaskResourceBinding points to the PipelineResource that will be used for the Task input or output called Name.

Deprecated: Unused, preserved only for backwards compatibility

▼ name `string`

Name is the name of the PipelineResource in the Pipeline's declaration

▼ paths `[]string`

Paths will probably be removed in #1284, and then PipelineResourceBinding can be used instead. The optional Path field corresponds to a path on disk at which the Resource can be found (used when providing the resource via mounted volume, overriding the default logic to fetch the Resource).

▼ resourceRef `object`

ResourceRef is a reference to the instance of the actual PipelineResource that should be used

▼ apiVersion `string`

API version of the referent

▼ **name** `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ **resourceSpec** `object`

ResourceSpec is specification of a resource that should be created and consumed by the task

▼ **description** `string`

Description is a user-facing description of the resource that may be used to populate a UI.

▼ **params** `[]object` required

ResourceParam declares a string value to use for the parameter called Name, and is used in the specific context of PipelineResources.

Deprecated: Unused, preserved only for backwards compatibility

▼ **name** `string` required

▼ **value** `string` required

▼ **secrets** `[]object`

SecretParam indicates which secret can be used to populate a field of the resource

Deprecated: Unused, preserved only for backwards compatibility

▼ **fieldName** **string** required

▼ **secretKey** **string** required

▼ **secretName** **string** required

▼ **type** **string** required

PipelineResourceType represents the type of endpoint the pipelineResource is, so that the controller will know this pipelineResource shouldx be fetched and optionally what additional metatdata should be provided for it.

Deprecated: Unused, preserved only for backwards compatibility

▼ **retries** **integer**

Retries represents how many times this TaskRun should be retried in the event of Task failure.

▼ **serviceAccountName** **string**

▼ **sidecarOverrides** **[]object**

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

▼ **name** `string` required

The name of the Sidecar to override.

▼ **resources** `object` required

The resource requirements to apply to the Sidecar.

▼ **claims** `[]object`

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ **name** `string` required

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.

▼ **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 **string**

Used for cancelling a TaskRun (and maybe more later on)

▼ statusMessage **string**

Status message for cancellation.

▼ stepOverrides **[]object**

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

▼ name **string** **required**

The name of the Step to override.

▼ resources **object** **required**

The resource requirements to apply to the Step.

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** required

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.

▼ 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/> ↗

▼ taskRef **object**

no more than one of the TaskRef and TaskSpec may be specified.

▼ apiVersion **string**

API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task

▼ **bundle** `string`

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.

▼ **kind** `string`

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

▼ **name** `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ **params** `[]object`

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

▼ **name** `string` required

▼ **value** required

▼ resolver `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ taskSpec

Specifying TaskSpec can be disabled by setting `disable-inline-spec` feature flag. See Task.spec (API version: tekton.dev/v1beta1)

▼ timeout `string`

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> ↗

▼ workspaces `[]object`

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

▼ configMap `object`

ConfigMap represents a configMap that should populate this workspace.

▼ 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 `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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` required

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 '..'.

▼ 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

▼ csi **object**

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

▼ driver **string** **required**

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.

▼ fsType **string**

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.

▼ nodePublishSecretRef **object**

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.

▼ 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> ↗

▼ readOnly `boolean`

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes `object`

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

▼ emptyDir `object`

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.

▼ medium `string`

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> ↗

▼ 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> ↗

▼ **name** string required

Name is the name of the workspace populated by the volume.

▼ **persistentVolumeClaim** object

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

▼ **claimName** string required

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> ↗

▼ **readOnly** boolean

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ **projected** object

Projected represents a projected volume that should populate this workspace.

▼ **defaultMode** integer

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.

▼ sources `[]object`

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

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels `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.

▼ 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** **required**

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap **object**

configMap information about the configMap data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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** required

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 '..'.

▼ 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

▼ downwardAPI **object**

downwardAPI information about the downwardAPI data to project

▼ items []object

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef object

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

▼ apiVersion string

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath string required

Path of the field to select in the specified API version.

▼ 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

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 '..'

▼ resourceFieldRef **object**

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

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secret **object**

secret information about the secret data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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** **required**

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 '..'.

▼ 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 field specify whether the Secret or its key must be defined

▼ **serviceAccountToken** **object**

serviceAccountToken is information about the serviceAccountToken data to project

▼ **audience** **string**

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.

▼ **expirationSeconds** **integer**

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.

▼ **path** **string** **required**

path is the path relative to the mount point of the file to project the token into.

▼ secret **object**

Secret represents a secret that should populate this workspace.

▼ 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 **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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 required

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 '..'.

▼ 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>

▼ 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).

▼ volumeClaimTemplate

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)

▼ status object

TaskRunStatus defines the observed state of TaskRun

▼ 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.

▼ cloudEvents **[]object**

CloudEventDelivery is the target of a cloud event along with the state of delivery.

▼ status **object**

CloudEventDeliveryState reports the state of a cloud event to be sent.

▼ condition **string**

Current status

▼ message **string** required

Error is the text of error (if any)

▼ retryCount **integer** required

RetryCount is the number of attempts of sending the cloud event

▼ sentAt **string**

SentAt is the time at which the last attempt to send the event was made

▼ target `string`

Target points to an addressable

▼ completionTime `string`

CompletionTime is the time the build completed.

▼ conditions `[]object`

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> ↗

▼ 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.

▼ status **string** required

Status of the condition, one of True, False, Unknown.

▼ type **string** required

Type of condition.

▼ observedGeneration **integer**

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ podName **string** required

PodName is the name of the pod responsible for executing this task's steps.

▼ provenance **object**

Provenance contains some key authenticated metadata about how a software artifact was built (what sources, what inputs/outputs, etc.).

▼ configSource **object**

Deprecated: Use RefSource instead

▼ digest **object**

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1":

"f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint **string**

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"

▼ uri **string**

URI indicates the identity of the source of the build definition. Example: ["https://github.com/tektoncd/catalog"](https://github.com/tektoncd/catalog)

▼ featureFlags **object**

FeatureFlags identifies the feature flags that were used during the task/pipeline run

▼ awaitSidecarReadiness **boolean**

▼ coschedule **string**

▼ disableCredsInit **boolean**

▼ disableInlineSpec **string**

▼ enableAPIFields **string**

▼ enableArtifacts **boolean**

▼ enableCELInWhenExpression **boolean**

▼ **enableConciseResolverSyntax** `boolean`

▼ **enableKeepPodOnCancel** `boolean`

▼ **enableKubernetesSidecar** `boolean`

▼ **enableParamEnum** `boolean`

▼ **enableProvenanceInStatus** `boolean`

▼ **enableStepActions** `boolean`

EnableStepActions is a no-op flag since StepActions are stable

▼ **enforceNonfalsifiability** `string`

▼ **maxResultSize** `integer`

▼ **requireGitSSHSecretKnownHosts** `boolean`

▼ **resultExtractionMethod** `string`

▼ **runningInEnvWithInjectedSidecars** `boolean`

▼ sendCloudEventsForRuns **boolean****▼ setSecurityContext** **boolean****▼ setSecurityContextReadOnlyRootFilesystem** **boolean****▼ verificationNoMatchPolicy** **string**

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

▼ refSource **object**

RefSource identifies the source where a remote task/pipeline came from.

▼ digest **object**

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1":

"f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint **string**

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"

▼ uri **string**

URI indicates the identity of the source of the build definition. Example: ["https://github.com/tektoncd/catalog"](https://github.com/tektoncd/catalog)

▼ resourcesResult **[]object**

RunResult is used to write key/value pairs to TaskRun pod termination messages. The key/value pairs may come from the entrypoint binary, or represent a TaskRunResult. If they represent a TaskRunResult, the key is the name of the result and the value is the JSON-serialized value of the result.

▼ key **string** required

▼ resourceName **string**

ResourceName may be used in tests, but it is not populated in termination messages. It is preserved here for backwards compatibility and will not be ported to v1.

▼ type **integer**

ResultType used to find out whether a RunResult is from a task result or not Note that ResultsType is another type which is used to define the data type (e.g. string, array, etc) we used for Results

▼ value **string** required

▼ retriesStatus

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. See TaskRun.status (API version: tekton.dev/v1beta1)

▼ sidecars []object

SidecarState reports the results of running a sidecar in a Task.

▼ container string

▼ imageID string

▼ name string

▼ running object

Details about a running container

▼ startedAt string

Time at which the container was last (re-)started

▼ terminated object

Details about a terminated container

▼ containerID **string**

Container's ID in the format '://<container_id>'

▼ exitCode **integer** **required**

Exit status from the last termination of the container

▼ finishedAt **string**

Time at which the container last terminated

▼ message **string**

Message regarding the last termination of the container

▼ reason **string**

(brief) reason from the last termination of the container

▼ signal **integer**

Signal from the last termination of the container

▼ startedAt **string**

Time at which previous execution of the container started

▼ waiting **object**

Details about a waiting container

▼ message `string`

Message regarding why the container is not yet running.

▼ reason `string`

(brief) reason the container is not yet running.

▼ spanContext `object`

SpanContext contains tracing span context fields

▼ startTime `string`

StartTime is the time the build is actually started.

▼ steps `[]object`

StepState reports the results of running a step in a Task.

▼ container `string`**▼ imageID** `string`**▼ inputs** `[]object`

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

▼ buildOutput `boolean`

Indicate if the artifact is a build output or a by-product

▼ **name** `string`

The artifact's identifying category name

▼ **values** `[]object`

ArtifactValue represents a specific value or data element within an Artifact.

▼ **digest** `object`

▼ **uri** `string`

▼ **name** `string`

▼ **outputs** `[]object`

Artifact represents an artifact within a system, potentially containing multiple values associated with it.

▼ **buildOutput** `boolean`

Indicate if the artifact is a build output or a by-product

▼ **name** `string`

The artifact's identifying category name

▼ values **[]object**

ArtifactValue represents a specific value or data element within an Artifact.

▼ digest **object****▼ uri** **string****▼ provenance** **object**

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.

▼ configSource **object**

Deprecated: Use RefSource instead

▼ digest **object**

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint **string**

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"

▼ uri **string**

URI indicates the identity of the source of the build definition.

Example: "<https://github.com/tektoncd/catalog>"

▼ featureFlags **object**

FeatureFlags identifies the feature flags that were used during the task/pipeline run

▼ awaitSidecarReadiness **boolean****▼ coschedule** **string****▼ disableCredsInit** **boolean****▼ disableInlineSpec** **string****▼ enableAPIFields** **string****▼ enableArtifacts** **boolean****▼ enableCELInWhenExpression** **boolean****▼ enableConciseResolverSyntax** **boolean**

▼ **enableKeepPodOnCancel** `boolean`

▼ **enableKubernetesSidecar** `boolean`

▼ **enableParamEnum** `boolean`

▼ **enableProvenanceInStatus** `boolean`

▼ **enableStepActions** `boolean`

EnableStepActions is a no-op flag since StepActions are stable

▼ **enforceNonfalsifiability** `string`

▼ **maxResultSize** `integer`

▼ **requireGitSSHSecretKnownHosts** `boolean`

▼ **resultExtractionMethod** `string`

▼ **runningInEnvWithInjectedSidecars** `boolean`

▼ **sendCloudEventsForRuns** `boolean`

▼ setSecurityContext `boolean`**▼ setSecurityContextReadOnlyRootFilesystem**`boolean`**▼ verificationNoMatchPolicy** `string`

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

▼ refSource `object`

RefSource identifies the source where a remote task/pipeline came from.

▼ digest `object`

Digest is a collection of cryptographic digests for the contents of the artifact specified by URI. Example: {"sha1": "f99d13e554ffcb696dee719fa85b695cb5b0f428"}

▼ entryPoint `string`

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"

▼ uri `string`

URI indicates the identity of the source of the build definition.

Example: "<https://github.com/tektoncd/catalog> ^"

▼ results `[]object`

TaskRunResult used to describe the results of a task

▼ name `string` required

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 required

Value the given value of the result

▼ running `object`

Details about a running container

▼ startedAt `string`

Time at which the container was last (re-)started

▼ terminated **object**

Details about a terminated container

▼ containerID **string**

Container's ID in the format '://<container_id>'

▼ exitCode **integer** required

Exit status from the last termination of the container

▼ finishedAt **string**

Time at which the container last terminated

▼ message **string**

Message regarding the last termination of the container

▼ reason **string**

(brief) reason from the last termination of the container

▼ signal **integer**

Signal from the last termination of the container

▼ startedAt **string**

Time at which previous execution of the container started

▼ waiting **object**

Details about a waiting container

▼ message **string**

Message regarding why the container is not yet running.

▼ reason **string**

(brief) reason the container is not yet running.

▼ taskResults **[]object**

TaskRunResult used to describe the results of a task

▼ name **string** required

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 required

Value the given value of the result

▼ taskSpec

TaskSpec contains the Spec from the dereferenced Task definition used to instantiate this TaskRun. See Task.spec (API version tekton.dev/v1beta1)



ClusterTask [tekton.dev/v1]

tekton.dev group

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.

v1beta1 version

▼ spec object

Spec holds the desired state of the Task from the client

▼ description string

Description is a user-facing description of the task that may be used to populate a UI.

▼ displayName string

DisplayName is a user-facing name of the task that may be used to populate a UI.

▼ params []object

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

▼ default object

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name `string` required

Name declares the name by which a parameter is referenced.

▼ properties `object`

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

▼ type `string`

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ resources `object`

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

▼ inputs `[]object`

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

▼ description `string`

Description is a user-facing description of the declared resource that may be used to populate a UI.

▼ name `string` required

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.

▼ optional `boolean`

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)

▼ targetPath `string`

TargetPath is the path in workspace directory where the resource will be copied.

▼ type `string` required

Type is the type of this resource;

▼ outputs `[]object`

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

▼ description `string`

Description is a user-facing description of the declared resource that may be used to populate a UI.

▼ name `string` required

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.

▼ optional `boolean`

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)

▼ targetPath `string`

TargetPath is the path in workspace directory where the resource will be copied.

▼ **type** `string` required

Type is the type of this resource;

▼ **results** `[]object`

TaskResult used to describe the results of a task

▼ **description** `string`

Description is a human-readable description of the result

▼ **name** `string` required

Name the given name

▼ **properties** `object`

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

▼ **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** `object`

Value the expression used to retrieve the value of the result from an underlying Step.

▼ sidecars []object

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

▼ args []string

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> ↗

▼ command []string

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> ↗

▼ env []object

EnvVar represents an environment variable present in a Container.

▼ name string required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** **required**

The key to select.

▼ name **string**

Name of the referent. 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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** **required**

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secretKeyRef **object**

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

▼ key **string** required

The key of the secret to select from. Must be a valid secret key.

▼ name **string**

Name of the referent. 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

▼ envFrom **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ name string

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ optional boolean

Specify whether the ConfigMap must be defined

▼ prefix string

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef object

The Secret to select from

▼ name string

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ optional boolean

Specify whether the Secret must be defined

▼ 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, IfNotPresent. Defaults to Always if :latest tag is specified, or IfNotPresent otherwise. Cannot be updated. More info:

<https://kubernetes.io/docs/concepts/containers/images#updating-images> ↗

▼ **lifecycle** `object`

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

▼ **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> ↗

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **sleep** `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ 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> ↗

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ sleep **object**

Sleep represents the duration that the container should sleep before being terminated.

▼ seconds **integer** required

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

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.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ livenessProbe **object**

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>



▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies an action involving a GRPC port.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** 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> ↗

▼ **periodSeconds** integer

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** object

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** `required`

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ name **string** required

Name of the Sidecar specified as a DNS_LABEL. Each Sidecar in a Task must have a unique name (DNS_LABEL). Cannot be updated.

▼ ports **[]object**

ContainerPort represents a network port in a single container.

▼ containerPort **integer** required

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.

▼ protocol **string**

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ 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/pod-lifecycle#container-probes>



▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies an action involving a GRPC port.

▼ port **integer** required

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.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** `required`

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies an action involving a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ resources **object**

Compute Resources required by this Sidecar. Cannot be updated. More info:

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** required

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.

▼ 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/> ↗

▼ restartPolicy `string`

RestartPolicy refers to kubernetes RestartPolicy. It can only be set for an initContainer and must have it's policy set to "Always". It is currently left optional to help support Kubernetes versions prior to 1.29 when this feature was introduced.

▼ script `string`

Script is the contents of an executable file to execute.

If Script is not empty, the Step cannot have an Command or Args.

▼ securityContext `object`

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/> ↗

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ 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.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **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.

▼ **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.

▼ 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.

▼ 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.

▼ **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.

▼ **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` `required`

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.

▼ windowsOptions **object**

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.

▼ 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.

▼ 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 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.

▼ startupProbe `object`

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>

**▼ exec** `object`

Exec specifies the action to take.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ **grpc** **object**

GRPC specifies an action involving a GRPC port.

▼ **port** **integer** required

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.

▼ **httpGet** **object**

HTTPGet specifies the http request to perform.

▼ **host** **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ initialDelaySeconds `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> ↗

▼ periodSeconds `integer`

How often (in seconds) to perform the probe. Default to 10 seconds.

Minimum value is 1.

▼ successThreshold `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket `object`

TCPSocket specifies an action involving a TCP port.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port `required`

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.

▼ terminationGracePeriodSeconds `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ stdin boolean

Whether this Sidecar should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the Sidecar will always result in EOF. Default is false.

▼ stdinOnce boolean

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true the stdin stream will remain open across multiple attach sessions. If stdinOnce is set to true, stdin is opened on Sidecar start, is empty until the first client attaches to stdin, and then remains open and accepts data until the client disconnects, at which time stdin is closed and remains closed until the Sidecar is restarted. If this flag is false, a container processes that reads from stdin will never receive an EOF. Default is false

▼ terminationMessagePath string

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, such as an assertion failure message. Will be truncated by the

node 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.

▼ **terminationMessagePolicy** `string`

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 Sidecar log output if the termination message file is empty and the Sidecar exited with an error. The log output is limited to 2048 bytes or 80 lines, whichever is smaller. Defaults to File. Cannot be updated.

▼ **tty** `boolean`

Whether this Sidecar should allocate a TTY for itself, also requires 'stdin' to be true. Default is false.

▼ **volumeDevices** `[]object`

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

▼ **devicePath** `string` required

devicePath is the path inside of the container that the device will be mapped to.

▼ **name** `string` required

name must match the name of a persistentVolumeClaim in the pod

▼ **volumeMounts** `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain '.'.

▼ mountPropagation `string`

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.

▼ name `string` required

This must match the Name of a Volume.

▼ readOnly `boolean`

Mounted read-only if true, read-write otherwise (false or unspecified). Defaults to false.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ workingDir `string`

Sidecar'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.

▼ workspaces `[]object`

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

▼ mountPath `string` required

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.

▼ name `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ stepTemplate `object`

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

▼ args `[]string`

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> ↗

▼ **command** []string

Entrypoint array. Not executed within a shell. The docker 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> ↗

▼ **env** []object

EnvVar represents an environment variable present in a Container.

▼ **name** string required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** **required**

The key to select.

▼ name **string**

Name of the referent. 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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** **string** required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** **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.

▼ **containerName** **string**

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** **string** required

Required: resource to select

▼ **secretKeyRef** **object**

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

▼ key **string** **required**

The key of the secret to select from. Must be a valid secret key.

▼ name **string**

Name of the referent. 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

▼ envFrom **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ name **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ optional **boolean**

Specify whether the ConfigMap must be defined

▼ **prefix** `string`

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ **secretRef** `object`

The Secret to select from

▼ **name** `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ **optional** `boolean`

Specify whether the Secret must be defined

▼ **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 like 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#updating-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.

▼ 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> ↗

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ **host** `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ **httpHeaders** `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

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

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ sleep `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ seconds `integer` required

Seconds is the number of seconds to sleep.

▼ tcpSocket `object`

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.

▼ host `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ 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> ↗

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ sleep **object**

Sleep represents the duration that the container should sleep before being terminated.

▼ seconds **integer** required

Seconds is the number of seconds to sleep.

▼ 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.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ 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/pod-lifecycle#container-probes>



Deprecated: This field will be removed in a future release.

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies an action involving a GRPC port.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** 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> ↗

▼ **periodSeconds** integer

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** object

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** `required`

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ name `string` required

Default name for each Step specified as a DNS_LABEL. Each Step in a Task must have a unique name. Cannot be updated.

Deprecated: This field will be removed in a future release.

▼ ports `[]object`

ContainerPort represents a network port in a single container.

▼ containerPort `integer` required

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.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe **object**

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.

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies an action involving a GRPC port.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ initialDelaySeconds **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> ↗

▼ periodSeconds **integer**

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ successThreshold **integer**

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies an action involving a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ resources **object**

Compute Resources required by this Step. Cannot be updated. More info:

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** **required**

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.

▼ 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/> ↗

▼ 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/> ↗

▼ allowPrivilegeEscalation boolean

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ 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.

▼ add []string

Added capabilities

▼ drop []string

Removed capabilities

▼ **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.

▼ **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.

▼ 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.

▼ 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.

▼ 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.

▼ 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** **required**

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.

▼ windowsOptions **object**

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.

▼ gmsaCredentialSpec `string`

GMSACredentialSpec is where the GMSA admission webhook ([https://github.com/kubernetes-sigs/windows-gmsa ↗](https://github.com/kubernetes-sigs/windows-gmsa)) inlines the contents of the GMSA credential spec named by the `GMSACredentialSpecName` 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 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.

▼ startupProbe `object`

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.

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies an action involving a GRPC port.

▼ port **integer** **required**

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.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** `required`

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.

Minimum value is 1.

▼ tcpSocket **object**

TCPSocket specifies an action involving a TCP port.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port **required**

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.

▼ terminationGracePeriodSeconds **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ 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/pods/pod-lifecycle#container-probes> ↗

▼ **stdin** **boolean**

Whether this Step should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the Step will always result in EOF. Default is false.

Deprecated: This field will be removed in a future release.

▼ **stdinOnce** **boolean**

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true 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, and then 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

Deprecated: This field will be removed in a future release.

▼ **terminationMessagePath** **string**

Deprecated: This field will be removed in a future release and cannot be meaningfully used.

▼ **terminationMessagePolicy** **string**

Deprecated: This field will be removed in a future release and cannot be meaningfully used.

▼ tty **boolean**

Whether this Step should allocate a DeprecatedTTY for itself, also requires 'stdin' to be true. Default is false.

Deprecated: This field will be removed in a future release.

▼ volumeDevices **[]object**

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

▼ devicePath **string** **required**

devicePath is the path inside of the container that the device will be mapped to.

▼ name **string** **required**

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts **[]object**

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath **string** **required**

Path within the container at which the volume should be mounted. Must not contain '..'.

▼ mountPropagation **string**

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.

▼ **name** `string` required

This must match the Name of a Volume.

▼ **readOnly** `boolean`

Mounted read-only if true, read-write otherwise (false or unspecified).

Defaults to false.

▼ **subPath** `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ **subPathExpr** `string`

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.

▼ **workingDir** `string`

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.

▼ steps `[]object`

Step runs a subcomponent of a Task

▼ args `[]string`

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> ↗

▼ command `[]string`

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> ↗

▼ env `[]object`

EnvVar represents an environment variable present in a Container.

▼ name `string` required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** **required**

The key to select.

▼ name **string**

Name of the referent. 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

▼ fieldRef **object**

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.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** **required**

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secretKeyRef **object**

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

▼ key **string** required

The key of the secret to select from. Must be a valid secret key.

▼ name **string**

Name of the referent. 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

▼ envFrom **[]object**

EnvFromSource represents the source of a set of ConfigMaps

▼ configMapRef **object**

The ConfigMap to select from

▼ name string

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ optional boolean

Specify whether the ConfigMap must be defined

▼ prefix string

An optional identifier to prepend to each key in the ConfigMap. Must be a C_IDENTIFIER.

▼ secretRef object

The Secret to select from

▼ name string

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ optional boolean

Specify whether the Secret must be defined

▼ image string

Image reference name to run for this Step. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ **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#updating-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.

▼ **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> ↗

▼ **exec** `object`

Exec specifies the action to take.

▼ **command** `[]string`

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ port required

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.

▼ **sleep** `object`

Sleep represents the duration that the container should sleep before being terminated.

▼ **seconds** `integer` required

Seconds is the number of seconds to sleep.

▼ **tcpSocket** `object`

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.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** required

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.

▼ 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> ↗

▼ exec **object**

Exec specifies the action to take.

▼ command **[]string**

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ sleep **object**

Sleep represents the duration that the container should sleep before being terminated.

▼ seconds **integer** required

Seconds is the number of seconds to sleep.

▼ tcpSocket **object**

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.

▼ host **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ port required

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.

▼ livenessProbe **object**

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.

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc `object`

GRPC specifies an action involving a GRPC port.

▼ port `integer` required

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host **string**

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders **[]object**

HTTPHeader describes a custom header to be used in HTTP probes

▼ name **string** required

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

▼ value **string** required

The header field value

▼ path **string**

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** 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> ↗

▼ **periodSeconds** integer

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** integer

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** object

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** `required`

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ name **string** required

Name of the Step specified as a DNS_LABEL. Each Step in a Task must have a unique name.

▼ onError **string**

OnError defines the exiting behavior of a container on error can be set to [continue | stopAndFail]

▼ params **[]object**

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

▼ name **string** required**▼ value** **object** required

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.

▼ ports **[]object**

ContainerPort represents a network port in a single container.

▼ containerPort **integer** required

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.

▼ protocol `string`

Protocol for port. Must be UDP, TCP, or SCTP. Defaults to "TCP".

▼ readinessProbe `object`

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.

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

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.

▼ failureThreshold **integer**

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc **object**

GRPC specifies an action involving a GRPC port.

▼ port **integer** required

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.

▼ httpGet **object**

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ name `string` required

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

▼ value `string` required

The header field value

▼ path `string`

Path to access on the HTTP server.

▼ port required

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.

▼ **initialDelaySeconds** **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> ↗

▼ **periodSeconds** **integer**

How often (in seconds) to perform the probe. Default to 10 seconds. Minimum value is 1.

▼ **successThreshold** **integer**

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup. Minimum value is 1.

▼ **tcpSocket** **object**

TCPSocket specifies an action involving a TCP port.

▼ **host** **string**

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** **required**

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.

▼ **terminationGracePeriodSeconds** **integer**

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **ref** **object**

Contains the reference to an existing StepAction.

▼ **name** **string**

Name of the referenced step

▼ params **[]object**

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

▼ name **string** **required****▼ value** **object** **required**

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.

▼ resolver **string**

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ resources **object**

Compute Resources required by this Step. Cannot be updated. More info:

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

▼ claims **[]object**

ResourceClaim references one entry in PodSpec.ResourceClaims.

▼ name **string** **required**

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.

▼ 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/> ↗

▼ results **[]object**

StepResult used to describe the Results of a Step.

This field is at a BETA stability level and gated by "enable-step-actions" feature flag.

▼ description **string**

Description is a human-readable description of the result

▼ name **string** **required**

Name the given name

▼ properties **object**

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

▼ type `string`

The possible types are 'string', 'array', and 'object', with 'string' as the default.

▼ 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/> ↗

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ 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.

▼ add `[]string`

Added capabilities

▼ drop `[]string`

Removed capabilities

▼ 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.

▼ 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.

▼ 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.

▼ 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.

▼ 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.

▼ 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` `required`

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.

▼ windowsOptions **object**

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.

▼ 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.

▼ 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 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.

▼ startupProbe `object`

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.

▼ exec `object`

Exec specifies the action to take.

▼ command `[]string`

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.

▼ failureThreshold `integer`

Minimum consecutive failures for the probe to be considered failed after having succeeded. Defaults to 3. Minimum value is 1.

▼ grpc `object`

GRPC specifies an action involving a GRPC port.

▼ port `integer` `required`

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.

▼ httpGet `object`

HTTPGet specifies the http request to perform.

▼ host `string`

Host name to connect to, defaults to the pod IP. You probably want to set "Host" in httpHeaders instead.

▼ httpHeaders `[]object`

HTTPHeader describes a custom header to be used in HTTP probes

▼ **name** `string` required

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

▼ **value** `string` required

The header field value

▼ **path** `string`

Path to access on the HTTP server.

▼ **port** required

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.

▼ **initialDelaySeconds** `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> ↗

▼ **periodSeconds** `integer`

How often (in seconds) to perform the probe. Default to 10 seconds.
Minimum value is 1.

▼ **successThreshold** `integer`

Minimum consecutive successes for the probe to be considered successful after having failed. Defaults to 1. Must be 1 for liveness and startup.
Minimum value is 1.

▼ **tcpSocket** `object`

TCPSocket specifies an action involving a TCP port.

▼ **host** `string`

Optional: Host name to connect to, defaults to the pod IP.

▼ **port** `required`

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.

▼ **terminationGracePeriodSeconds** `integer`

Optional duration in seconds the pod needs to terminate gracefully 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. Set this value longer than the expected cleanup time for your process. 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. The value zero indicates stop immediately via the kill signal (no opportunity to shut down). This is a beta field and requires enabling ProbeTerminationGracePeriod feature gate. Minimum value is 1. spec.terminationGracePeriodSeconds is used if unset.

▼ **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/pods/pod-lifecycle#container-probes> ↗

▼ **stderrConfig** `object`

Stores configuration for the stderr stream of the step.

▼ **path** `string`

Path to duplicate stdout stream to on container's local filesystem.

▼ **stdin** `boolean`

Whether this container should allocate a buffer for stdin in the container runtime. If this is not set, reads from stdin in the container will always result in EOF. Default is false.

Deprecated: This field will be removed in a future release.

▼ **stdinOnce** `boolean`

Whether the container runtime should close the stdin channel after it has been opened by a single attach. When stdin is true 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, and then 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

Deprecated: This field will be removed in a future release.

▼ stdoutConfig **object**

Stores configuration for the stdout stream of the step.

▼ path **string**

Path to duplicate stdout stream to on container's local filesystem.

▼ terminationMessagePath **string**

Deprecated: This field will be removed in a future release and can't be meaningfully used.

▼ terminationMessagePolicy **string**

Deprecated: This field will be removed in a future release and can't be meaningfully used.

▼ timeout **string**

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> ↗

▼ tty **boolean**

Whether this container should allocate a DeprecatedTTY for itself, also requires 'stdin' to be true. Default is false.

Deprecated: This field will be removed in a future release.

▼ volumeDevices **[]object**

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

▼ devicePath **string** **required**

devicePath is the path inside of the container that the device will be mapped to.

▼ name **string** **required**

name must match the name of a persistentVolumeClaim in the pod

▼ volumeMounts **[]object**

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath **string** **required**

Path within the container at which the volume should be mounted. Must not contain '..'.

▼ mountPropagation **string**

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.

▼ **name** `string` required

This must match the Name of a Volume.

▼ **readOnly** `boolean`

Mounted read-only if true, read-write otherwise (false or unspecified).

Defaults to false.

▼ **subPath** `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ **subPathExpr** `string`

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.

▼ **when** `[]object`

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

▼ **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> ↗

▼ input `string`

Input is the string for guard checking which can be a static input or an output from a parent Task

▼ operator `string`

Operator that represents an Input's relationship to the values

▼ values `[]string`

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

▼ workingDir `string`

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.

▼ workspaces `[]object`

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

▼ mountPath `string` required

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.

▼ **name** `string` required

Name is the name of the workspace this Step or Sidecar wants access to.

▼ **volumes** `[]object`

Volume represents a named volume in a pod that may be accessed by any container in the pod.

▼ **awsElasticBlockStore** `object`

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> ↗

▼ **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#awselasticblockstore>



▼ **partition** `integer`

partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty).

▼ readOnly `boolean`

readOnly value true will force the readOnly setting in VolumeMounts. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#awselasticsearchblockstore>

**▼ volumeID** `string` required

volumeID is unique ID of the persistent disk resource in AWS (Amazon EBS volume). More info:

<https://kubernetes.io/docs/concepts/storage/volumes#awselasticsearchblockstore>

**▼ azureDisk** `object`

azureDisk represents an Azure Data Disk mount on the host and bind mount to the pod.

▼ cachingMode `string`

cachingMode is the Host Caching mode: None, Read Only, Read Write.

▼ diskName `string` required

diskName is the Name of the data disk in the blob storage

▼ diskURI `string` required

diskURI is the URI of data disk in the blob storage

▼ fsType `string`

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.

▼ **kind** `string`

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

▼ **readOnly** `boolean`

readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **azureFile** `object`

azureFile represents an Azure File Service mount on the host and bind mount to the pod.

▼ **readOnly** `boolean`

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **secretName** `string` required

secretName is the name of secret that contains Azure Storage Account Name and Key

▼ **shareName** `string` required

shareName is the azure share Name

▼ cephfs **object**

cephFS represents a Ceph FS mount on the host that shares a pod's lifetime

▼ monitors **[]string** required

monitors is Required: Monitors is a collection of Ceph monitors More info:

<https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it> ↗

▼ path **string**

path is Optional: Used as the mounted root, rather than the full Ceph tree, default is /

▼ readOnly **boolean**

readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info:

<https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it> ↗

▼ secretFile **string**

secretFile is Optional: SecretFile is the path to key ring for User, default is /etc/ceph/user.secret More info:

<https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it> ↗

▼ secretRef **object**

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> ↗

▼ name string

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ user string

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> ↗

▼ cinder object

cinder represents a cinder volume attached and mounted on kubelets host machine. More info: <https://examples.k8s.io/mysql-cinder-pd/README.md> ↗

▼ fsType string

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> ↗

▼ readOnly boolean

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> ↗

▼ secretRef object

secretRef is optional: points to a secret object containing parameters used to connect to OpenStack.

▼ name **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ volumeID **string** **required**

volumeID used to identify the volume in cinder. More info:

<https://examples.k8s.io/mysql-cinder-pd/README.md> ↗

▼ configMap **object**

configMap represents a configMap that should populate this volume

▼ 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 **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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** **required**

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 '..'.

▼ 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

▼ csi **object**

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

▼ driver **string** **required**

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.

▼ fsType `string`

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.

▼ nodePublishSecretRef `object`

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.

▼ name `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ readOnly `boolean`

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes `object`

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

▼ downwardAPI **object**

downwardAPI represents downward API about the pod that should populate this volume

▼ defaultMode **integer**

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.

▼ items **[]object**

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef **object**

Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath **string** required

Path of the field to select in the specified API version.

▼ 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

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 '..'

▼ resourceFieldRef **object**

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

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ emptyDir **object**

emptyDir represents a temporary directory that shares a pod's lifetime. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#emptydir> ↗

▼ medium **string**

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> ↗

▼ 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> ↗

▼ ephemeral **object**

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.

▼ volumeClaimTemplate object

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.

▼ metadata object

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** required

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.

▼ accessModes **[]string**

accessModes contains the desired access modes the volume should have. More info:

<https://kubernetes.io/docs/concepts/storage/persistent-volumes#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.

▼ apiGroup `string`

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.

▼ kind `string` `required`

Kind is the type of resource being referenced

▼ name `string` `required`

Name is the name of resource being referenced

▼ 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 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.

▼ `apiGroup` `string`

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.

▼ `kind` `string` required

Kind is the type of resource being referenced

▼ `name` `string` required

Name is the name of resource being referenced

▼ `namespace` `string`

Namespace is the namespace of resource being referenced Note that when a namespace is specified, a

gateway.networking.k8s.io/ReferenceGrant 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 CrossNamespaceVolumeDataSource feature gate to be enabled.

▼ resources **object**

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> ↗

▼ 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/> ↗

▼ selector **object**

selector is a label query over volumes to consider for binding.

▼ matchExpressions **[]object**

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

▼ key **string** required

key is the label key that the selector applies to.

▼ operator **string** required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values **[]string**

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.

▼ matchLabels **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.

▼ storageClassName **string**

storageClassName is the name of the StorageClass required by the claim. More info:

<https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1> ↗

▼ volumeAttributesClassName **string**

volumeAttributesClassName may be used to set the VolumeAttributesClass used by this claim. If specified, the CSI driver will create or update the volume with the 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 no VolumeAttributesClass will be applied to the claim but 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 set by the persistentvolume controller if it exists. If the resource referred to by volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending state, as reflected by the modifyVolumeStatus field, until such as a resource exists. More info:

<https://kubernetes.io/docs/concepts/storage/persistent->

[volumes#volumeattributesclass ↗](#) (Alpha) Using this field requires the VolumeAttributesClass feature gate to be enabled.

▼ **volumeMode** `string`

volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec.

▼ **volumeName** `string`

volumeName is the binding reference to the PersistentVolume backing this claim.

▼ **fc** `object`

fc represents a Fibre Channel resource that is attached to a kubelet's host machine and then exposed to the pod.

▼ **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.

▼ **lun** `integer`

lun is Optional: FC target lun number

▼ **readOnly** `boolean`

readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ **targetWWNs** `[]string`

targetWWNs is Optional: FC target worldwide names (WWNs)

▼ **wwids** `[]string`

wwids Optional: FC volume world wide identifiers (wwids) Either wwids or combination of targetWWNs and lun must be set, but not both simultaneously.

▼ **flexVolume** `object`

flexVolume represents a generic volume resource that is provisioned/attached using an exec based plugin.

▼ **driver** `string` required

driver is the name of the driver to use for this volume.

▼ **fsType** `string`

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". The default filesystem depends on FlexVolume script.

▼ **options** `object`

options is Optional: this field holds extra command options if any.

▼ readOnly `boolean`

readOnly is Optional: defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ secretRef `object`

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.

▼ name `string`

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ flocker `object`

flocker represents a Flocker volume attached to a kubelet's host machine. This depends on the Flocker control service being running

▼ datasetName `string`

datasetName is Name of the dataset stored as metadata -> name on the dataset for Flocker should be considered as deprecated

▼ datasetUUID `string`

datasetUUID is the UUID of the dataset. This is unique identifier of a Flocker dataset

▼ gcePersistentDisk **object**

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> ↗

▼ fsType **string**

fsType 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" if unspecified.

More info:

<https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk> ↗

▼ partition **integer**

partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty). More info:

<https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk> ↗

▼ pdName **string** required

pdName 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> ↗

▼ readOnly **boolean**

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk> ↗

▼ gitRepo **object**

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.

▼ directory **string**

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.

▼ repository **string** required

repository is the URL

▼ revision **string**

revision is the commit hash for the specified revision.

▼ glusterfs **object**

glusterfs represents a Glusterfs mount on the host that shares a pod's lifetime.

More info: <https://examples.k8s.io/volumes/glusterfs/README.md> ↗

▼ endpoints **string** required

endpoints is the endpoint name that details Glusterfs topology. More info:

<https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod> ↗

▼ **path** **string** required

path is the Glusterfs volume path. More info:

<https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod> ↗

▼ **readOnly** **boolean**

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> ↗

▼ **hostPath** **object**

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> ↗

▼ **path** **string** required

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> ↗

▼ iscsi **object**

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> ↗

▼ chapAuthDiscovery **boolean**

chapAuthDiscovery defines whether support iSCSI Discovery CHAP authentication

▼ chapAuthSession **boolean**

chapAuthSession defines whether support iSCSI Session CHAP authentication

▼ 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#iscsi> ↗

▼ initiatorName **string**

initiatorName is the custom iSCSI Initiator Name. If initiatorName is specified with iscsiInterface simultaneously, new iSCSI interface : will be created for the connection.

▼ iqn **string** required

iqn is the target iSCSI Qualified Name.

▼ iscsiInterface **string**

iscsiInterface is the interface Name that uses an iSCSI transport. Defaults to 'default' (tcp).

▼ lun **integer** required

lun represents iSCSI Target Lun number.

▼ portals **[]string**

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).

▼ readOnly **boolean**

readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false.

▼ secretRef **object**

secretRef is the CHAP Secret for iSCSI target and initiator authentication

▼ name **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ **targetPortal** **string** required

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).

▼ **name** **string** required

name of the volume. Must be a DNS_LABEL and unique within the pod. 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 pod's lifetime More info: <https://kubernetes.io/docs/concepts/storage/volumes#nfs> ↗

▼ **path** **string** required

path that is exported by the NFS server. More info: <https://kubernetes.io/docs/concepts/storage/volumes#nfs> ↗

▼ **readOnly** **boolean**

readOnly here will force the NFS export to be mounted with read-only permissions. Defaults to false. More info: <https://kubernetes.io/docs/concepts/storage/volumes#nfs> ↗

▼ **server** **string** required

server is the hostname or IP address of the 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> ↗

▼ claimName **string** required

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> ↗

▼ readOnly **boolean**

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ photonPersistentDisk **object**

photonPersistentDisk represents a PhotonController persistent disk attached and mounted on kubelets host machine

▼ 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** required

pdID is the ID that identifies Photon Controller persistent disk

▼ portworxVolume **object**

portworxVolume represents a portworx volume attached and mounted on kubelets host machine

▼ fsType **string**

fSType represents the filesystem type to mount Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs". Implicitly inferred to be "ext4" if unspecified.

▼ readOnly **boolean**

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ volumeID **string** required

volumeID uniquely identifies a Portworx volume

▼ projected **object**

projected items for all in one resources secrets, configmaps, and downward API

▼ defaultMode **integer**

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.

▼ sources `[]object`

Projection that may be projected along with other supported volume types

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels `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.

▼ 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** **required**

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap **object**

configMap information about the configMap data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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** required

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 '..'.

▼ **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

▼ **downwardAPI** **object**

downwardAPI information about the downwardAPI data to project

▼ **items** **[]object**

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ **fieldRef** **object**

Required: Selects a field of the pod: only annotations, labels, name and namespace are supported.

▼ **apiVersion** **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** **string** **required**

Path of the field to select in the specified API version.

▼ **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**

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 '..'

▼ resourceFieldRef object

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

▼ containerName string

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource string required

Required: resource to select

▼ secret object

secret information about the secret data to project

▼ items []object

Maps a string key to a path within a volume.

▼ **key** **string** required

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** required

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 '..'.

▼ **name** **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ **optional** **boolean**

optional field specify whether the Secret or its key must be defined

▼ serviceAccountToken **object**

serviceAccountToken is information about the serviceAccountToken data to project

▼ audience **string**

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.

▼ expirationSeconds **integer**

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.

▼ path **string** required

path is the path relative to the mount point of the file to project the token into.

▼ quobyte **object**

quobyte represents a Quobyte mount on the host that shares a pod's lifetime

▼ 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** required

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** required

volume is a string that references an already created Quobyte volume by name.

▼ rbd **object**

rbd 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> ↗

▼ 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** required

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 **[]string** required

monitors is a collection of Ceph monitors. More info: <https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it> ↗

▼ pool **string**

pool is the rados pool name. Default is rbd. More info: <https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it> ↗

▼ readOnly **boolean**

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> ↗

▼ secretRef **object**

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> ↗

▼ name **string**

Name of the referent. More info: <https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ user **string**

user is the rados user name. Default is admin. More info: <https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it> ↗

▼ scaleIO **object**

scaleIO represents a ScaleIO persistent volume attached and mounted on Kubernetes nodes.

▼ fsType **string**

fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Default is "xfs".

▼ gateway **string** required

gateway is the host address of the ScaleIO API Gateway.

▼ protectionDomain **string**

protectionDomain is the name of the ScaleIO Protection Domain for the configured storage.

▼ readOnly **boolean**

readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ secretRef **object** required

secretRef references to the secret for ScaleIO user and other sensitive information. If this is not provided, Login operation will fail.

▼ name **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ sslEnabled **boolean**

sslEnabled Flag enable/disable SSL communication with Gateway, default false

▼ storageMode **string**

storageMode indicates whether the storage for a volume should be ThickProvisioned or ThinProvisioned. Default is ThinProvisioned.

▼ **storagePool** `string`

storagePool is the ScaleIO Storage Pool associated with the protection domain.

▼ **system** `string` required

system is the name of the storage system as configured in ScaleIO.

▼ **volumeName** `string`

volumeName is the name of a volume already created in the ScaleIO system that is associated with this volume source.

▼ **secret** `object`

secret represents a secret that should populate this volume. More info:

<https://kubernetes.io/docs/concepts/storage/volumes#secret> ↗

▼ **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 `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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` required

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 '..'.

▼ 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> ↗

▼ storageos **object**

storageOS represents a StorageOS volume attached and mounted on Kubernetes nodes.

▼ 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.

▼ readOnly **boolean**

readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

▼ secretRef **object**

secretRef specifies the secret to use for obtaining the StorageOS API credentials. If not specified, default values will be attempted.

▼ name **string**

Name of the referent. More info:

<https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names> ↗

▼ volumeName **string**

volumeName is the human-readable name of the StorageOS volume. Volume names are only unique within a namespace.

▼ volumeNamespace `string`

volumeNamespace 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 VolumeName 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.

▼ vsphereVolume `object`

vsphereVolume represents a vSphere volume attached and mounted on kubelets host machine

▼ fsType `string`

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.

▼ storagePolicyID `string`

storagePolicyID is the storage Policy Based Management (SPBM) profile ID associated with the StoragePolicyName.

▼ storagePolicyName `string`

storagePolicyName is the storage Policy Based Management (SPBM) profile name.

▼ volumePath `string` required

volumePath is the path that identifies vSphere volume vmdk

▼ workspaces `[]object`

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

▼ description `string`

Description is an optional human readable description of this volume.

▼ mountPath `string`

MountPath overrides the directory that the volume will be made available at.

▼ name `string` required

Name is the name by which you can bind the volume at runtime.

▼ optional `boolean`

Optional marks a Workspace as not being required in TaskRuns. By default this field is false and so declared workspaces are required.

▼ readOnly `boolean`

ReadOnly dictates whether a mounted volume is writable. By default this field is false and so mounted volumes are writable.

Run [tekton.dev/v1]

tekton.dev group

Run represents a single execution of a Custom Task.

v1alpha1 version

▼ spec object

RunSpec defines the desired state of Run

▼ params []object

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

▼ name string required

▼ value required

▼ podTemplate object

PodTemplate holds pod specific configuration

▼ affinity

If specified, the pod's scheduling constraints. See Pod.spec.affinity (API version: v1)

▼ automountServiceAccountToken `boolean`

AutomountServiceAccountToken indicates whether pods running as this service account should have an API token automatically mounted.

▼ dnsConfig `object`

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

▼ nameservers `[]string`

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

▼ options `[]object`

PodDNSConfigOption defines DNS resolver options of a pod.

▼ name `string`

Name is this DNS resolver option's name. Required.

▼ value `string`

Value is this DNS resolver option's value.

▼ searches `[]string`

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.

▼ dnsPolicy **string**

Set DNS policy for the pod. Defaults to "ClusterFirst". Valid values are 'ClusterFirst', 'Default' or 'None'. DNS parameters given in DNSConfig will be merged with the policy selected with DNSPolicy.

▼ enableServiceLinks **boolean**

EnableServiceLinks indicates whether information about services should be injected into pod's environment variables, matching the syntax of Docker links. Optional: Defaults to true.

▼ env **[]object**

EnvVar represents an environment variable present in a Container.

▼ name **string** required

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.

▼ configMapKeyRef **object**

Selects a key of a ConfigMap.

▼ key **string** required

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

▼ fieldRef **object**

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`.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of,
defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env
vars

▼ **divisor**

Specifies the output format of the exposed resources,
defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ secretKeyRef `object`

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

▼ key `string` required

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

▼ hostAliases `[]object`

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

▼ hostnames `[]string`

Hostnames for the above IP address.

▼ **ip** `string` required

IP address of the host file entry.

▼ **hostNetwork** `boolean`

HostNetwork specifies whether the pod may use the node network namespace

▼ **imagePullSecrets** `[]object`

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

▼ **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> ↗

▼ **nodeSelector** `object`

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/> ↗

▼ **priorityClassName** `string`

If specified, indicates the pod's priority. "system-node-critical" and "system-cluster-critical" are two special keywords which indicate the highest priorities with the former being the highest priority. Any other name must be defined by creating a PriorityClass object with that name. If not specified, the pod priority will be default or zero if there is no default.

▼ runtimeClassName `string`

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 named class, the pod will not be run. If unset or empty, the "legacy" RuntimeClass will be used, which is an implicit 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.

▼ schedulerName `string`

SchedulerName specifies the scheduler to be used to dispatch the Pod

▼ securityContext

SecurityContext holds pod-level security attributes and common container settings. Optional: Defaults to empty. See type description for default values of each field. See Pod.spec.securityContext (API version: v1)

▼ tolerations `[]object`

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

▼ 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.

▼ topologySpreadConstraints `[]object`

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

▼ 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.

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ operator `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values `[]string`

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.

▼ matchLabels `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.

▼ **matchLabelKeys** `[]string`

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).

▼ **maxSkew** `integer` required

MaxSkew describes the degree to which pods may be unevenly distributed. When `whenUnsatisfiable=DoNotSchedule`, 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 `whenUnsatisfiable=ScheduleAnyway`, 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.

▼ **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:

- 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` required

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.

▼ whenUnsatisfiable `string` required

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

When `Unsatisfiable` 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 *more* imbalanced. It's a required field.

▼ volumes

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)

▼ ref `object`

`TaskRef` can be used to refer to a specific instance of a task.

▼ apiVersion `string`

API version of the referent Note: A Task with non-empty `APIVersion` and `Kind` is considered a Custom Task

▼ bundle `string`

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.

▼ kind `string`

`TaskKind` indicates the Kind of the Task:

- 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

▼ name `string`

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params `[]object`

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

▼ name `string` required

▼ value required

▼ resolver `string`

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ retries `integer`

Used for propagating retries count to custom tasks

▼ serviceAccountName `string`

▼ spec `object`

Spec is a specification of a custom task

▼ **apiVersion** `string`

▼ **kind** `string`

▼ **metadata** `object`

PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask

▼ **annotations** `object`

▼ **labels** `object`

▼ **spec** `object`

Spec is a specification of a custom task

▼ **status** `string`

Used for cancelling a run (and maybe more later on)

▼ **statusMessage** `string`

Status message for cancellation.

▼ **timeout** `string`

Time after which the custom-task times out. Refer Go's ParseDuration documentation for expected format: <https://golang.org/pkg/time/#ParseDuration> ↗

▼ workspaces `[]object`

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

▼ configMap `object`

ConfigMap represents a configMap that should populate this workspace.

▼ 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 `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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` required

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 '..'.

▼ **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

▼ **csi** `object`

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

▼ **driver** `string` required

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.

▼ fsType **string**

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.

▼ nodePublishSecretRef **object**

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.

▼ 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>

▼ readOnly **boolean**

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes **object**

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

▼ emptyDir **object**

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.

▼ medium **string**

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> ↗

▼ 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> ↗

▼ name **string** required

Name is the name of the workspace populated by the volume.

▼ persistentVolumeClaim **object**

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

▼ claimName `string` required

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> ↗

▼ readOnly `boolean`

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ projected `object`

Projected represents a projected volume that should populate this workspace.

▼ defaultMode `integer`

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.

▼ sources `[]object`

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

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions []object

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

▼ key string required

key is the label key that the selector applies to.

▼ operator string required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ values []string

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.

▼ matchLabels **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.

▼ 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** **required**

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap object

configMap information about the configMap data to project

▼ items []object

Maps a string key to a path within a volume.

▼ key string required

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 required

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

...

▼ 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

▼ downwardAPI **object**

downwardAPI information about the downwardAPI data to project

▼ items **[]object**

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef **object**

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

▼ apiVersion **string**

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** **string** **required**

Path of the field to select in the specified API version.

▼ **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**

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 '..'

▼ **resourceFieldRef** **object**

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

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** **required**

Required: resource to select

▼ secret **object**

secret information about the secret data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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` required

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 '..'.

▼ **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 field specify whether the Secret or its key must be defined

▼ **serviceAccountToken** `object`

serviceAccountToken is information about the serviceAccountToken data to project

▼ **audience** `string`

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.

▼ expirationSeconds `integer`

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.

▼ path `string` required

path is the path relative to the mount point of the file to project the token into.

▼ secret `object`

Secret represents a secret that should populate this workspace.

▼ 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 `[]object`

Maps a string key to a path within a volume.

▼ key `string` required

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` required

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 '..'.

▼ 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> ↗

▼ 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).

▼ volumeClaimTemplate

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)

▼ status `object`

RunStatus defines the observed state of Run

▼ 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.

▼ completionTime `string`

CompletionTime is the time the build completed.

▼ conditions `[]object`

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](#)**▼ 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.

▼ status `string` `required`

Status of the condition, one of True, False, Unknown.

▼ type `string` `required`

Type of condition.

▼ extraFields `object`

ExtraFields holds arbitrary fields provided by the custom task controller.

▼ observedGeneration integer

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ results []object

RunResult used to describe the results of a task

▼ name string required

Name the given name

▼ value string required

Value the given value of the result

▼ retriesStatus array

RetriesStatus contains the history of RunStatus, in case of a retry.

▼ startTime string

StartTime is the time the build is actually started.



CustomRun [tekton.dev/v1]

tekton.dev group

CustomRun represents a single execution of a Custom Task.

v1beta1 version

▼ spec object

CustomRunSpec defines the desired state of CustomRun

▼ customRef object

TaskRef can be used to refer to a specific instance of a task.

▼ apiVersion string

API version of the referent Note: A Task with non-empty APIVersion and Kind is considered a Custom Task

▼ bundle string

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.

▼ kind string

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

▼ name string

Name of the referent; More info: <http://kubernetes.io/docs/user-guide/identifiers#names> ↗

▼ params []object

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

▼ name string required**▼ value** required**▼ resolver** string

Resolver is the name of the resolver that should perform resolution of the referenced Tekton resource, such as "git".

▼ customSpec object

Spec is a specification of a custom task

▼ apiVersion string**▼ kind** string

▼ metadata **object**

PipelineTaskMetadata contains the labels or annotations for an EmbeddedTask

▼ annotations **object****▼ labels** **object****▼ spec** **object**

Spec is a specification of a custom task

▼ params **[]object**

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

▼ name **string** **required****▼ value** **required****▼ retries** **integer**

Used for propagating retries count to custom tasks

▼ serviceAccountName **string**

▼ status `string`

Used for cancelling a customrun (and maybe more later on)

▼ statusMessage `string`

Status message for cancellation.

▼ timeout `string`

Time after which the custom-task times out. Refer Go's ParseDuration documentation for expected format: <https://golang.org/pkg/time/#ParseDuration> ↗

▼ workspaces `[]object`

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

▼ configMap `object`

ConfigMap represents a configMap that should populate this workspace.

▼ 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 `[]object`

Maps a string key to a path within a volume.

▼ key **string** required

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** required

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 '..'.

▼ 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

▼ **csi** **object**

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

▼ **driver** **string** required

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.

▼ **fsType** **string**

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.

▼ **nodePublishSecretRef** **object**

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.

▼ **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> ↗

▼ **readOnly** **boolean**

readOnly specifies a read-only configuration for the volume. Defaults to false (read/write).

▼ volumeAttributes **object**

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

▼ emptyDir **object**

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.

▼ medium **string**

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> ↗

▼ 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> ↗

▼ name **string** required

Name is the name of the workspace populated by the volume.

▼ persistentVolumeClaim **object**

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

▼ claimName **string** required

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> ↗

▼ readOnly **boolean**

readOnly Will force the ReadOnly setting in VolumeMounts. Default false.

▼ projected **object**

Projected represents a projected volume that should populate this workspace.

▼ defaultMode **integer**

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.

▼ sources `[]object`

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

▼ clusterTrustBundle `object`

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.

▼ 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".

▼ matchExpressions `[]object`

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

▼ key `string` required

key is the label key that the selector applies to.

▼ **operator** `string` required

operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

▼ **values** `[]string`

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.

▼ **matchLabels** `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.

▼ **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** **required**

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 selected ClusterTrustBundles will be unified and deduplicated.

▼ configMap **object**

configMap information about the configMap data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** **required**

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 required

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 '..'.

▼ 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

▼ downwardAPI object

downwardAPI information about the downwardAPI data to project

▼ items []object

DownwardAPIVolumeFile represents information to create the file containing the pod field

▼ fieldRef object

Required: Selects a field of the pod: only annotations, labels, name, namespace and uid are supported.

▼ apiVersion string

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath string required

Path of the field to select in the specified API version.

▼ 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

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 '..'

▼ resourceFieldRef **object**

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

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** required

Required: resource to select

▼ secret **object**

secret information about the secret data to project

▼ items **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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** required

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 '..'.

▼ 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 field specify whether the Secret or its key must be defined

▼ **serviceAccountToken** **object**

serviceAccountToken is information about the serviceAccountToken data to project

▼ **audience** **string**

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.

▼ **expirationSeconds** **integer**

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.

▼ **path** **string** **required**

path is the path relative to the mount point of the file to project the token into.

▼ secret **object**

Secret represents a secret that should populate this workspace.

▼ 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 **[]object**

Maps a string key to a path within a volume.

▼ key **string** required

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` required

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 '..'.

▼ **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>

▼ **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).

▼ **volumeClaimTemplate**

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)

▼ **status** `object`

CustomRunStatus defines the observed state of CustomRun

▼ 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.

▼ completionTime `string`

CompletionTime is the time the build completed.

▼ conditions `[]object`

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> ↗

▼ 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.

▼ **status** `string` required

Status of the condition, one of True, False, Unknown.

▼ **type** `string` required

Type of condition.

▼ **extraFields**

ExtraFields holds arbitrary fields provided by the custom task controller.

▼ **observedGeneration** `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ **results** `[]object`

CustomRunResult used to describe the results of a task

▼ **name** `string` required

Name the given name

▼ **value** `string` required

Value the given value of the result

▼ retriesStatus

RetriesStatus contains the history of CustomRunStatus, in case of a retry. See CustomRun.status (API version: tekton.dev/v1beta1)

▼ startTime `string`

StartTime is the time the build is actually started.

StepAction [tekton.dev/v1]

tekton.dev group

StepAction represents the actionable components of Step. The Step can only reference it from the cluster or using remote resolution.

v1alpha1 version

▼ spec object

Spec holds the desired state of the Step from the client

▼ args []string

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>

▼ command []string

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 []object

EnvVar represents an environment variable present in a Container.

▼ name string required

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.

▼ configMapKeyRef object

Selects a key of a ConfigMap.

▼ key string required

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

▼ **fieldRef** `object`

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.

▼ **apiVersion** `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ **fieldPath** `string` required

Path of the field to select in the specified API version.

▼ **resourceFieldRef** `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.

▼ **containerName** `string`

Container name: required for volumes, optional for env vars

▼ **divisor**

Specifies the output format of the exposed resources, defaults to "1"

▼ **resource** `string` required

Required: resource to select

▼ **secretKeyRef** `object`

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

▼ **key** `string` required

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

▼ image `string`

Image reference name to run for this StepAction. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ params `[]object`

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources).

Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

▼ default

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name **string** required

Name declares the name by which a parameter is referenced.

▼ properties **object**

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

▼ type **string**

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ results **[]object**

StepResult used to describe the Results of a Step.

▼ description **string**

Description is a human-readable description of the result

▼ name **string** required

Name the given name

▼ properties **object**

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

▼ type **string**

The possible types are 'string', 'array', and 'object', with 'string' as the default.

▼ 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.

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the no_new_privs 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.

▼ appArmorProfile `object`

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.

▼ localhostProfile `string`

`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".

▼ **type** `string` required

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.

▼ **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.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **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 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 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.

▼ **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.

▼ **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.

▼ **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** required

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.

▼ windowsOptions **object**

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.

▼ 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.

▼ 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 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.

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain '..'.

▼ mountPropagation `string`

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` required

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.

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.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ **workingDir** `string`

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.

StepAction represents the actionable components of Step. The Step can only reference it from the cluster or using remote resolution.

v1beta1

version

▼ **spec** `object`

Spec holds the desired state of the Step from the client

▼ **args** `[]string`

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> ↗

▼ **command** `[]string`

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 []object

EnvVar represents an environment variable present in a Container.

▼ name string required

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.

▼ configMapKeyRef object

Selects a key of a ConfigMap.

▼ key `string` required

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

▼ fieldRef `object`

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.

▼ apiVersion `string`

Version of the schema the FieldPath is written in terms of, defaults to "v1".

▼ fieldPath `string` required

Path of the field to select in the specified API version.

▼ resourceFieldRef **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.

▼ containerName **string**

Container name: required for volumes, optional for env vars

▼ divisor

Specifies the output format of the exposed resources, defaults to "1"

▼ resource **string** **required**

Required: resource to select

▼ secretKeyRef **object**

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

▼ key **string** **required**

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

▼ image `string`

Image reference name to run for this StepAction. More info:

<https://kubernetes.io/docs/concepts/containers/images> ↗

▼ params `[]object`

ParamSpec defines arbitrary parameters needed beyond typed inputs (such as resources).

Parameter values are provided by users as inputs on a TaskRun or PipelineRun.

▼ default

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.

▼ description `string`

Description is a user-facing description of the parameter that may be used to populate a UI.

▼ enum `[]string`

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.

▼ name **string** required

Name declares the name by which a parameter is referenced.

▼ properties **object**

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

▼ type **string**

Type is the user-specified type of the parameter. The possible types are currently "string", "array" and "object", and "string" is the default.

▼ results **[]object**

StepResult used to describe the Results of a Step.

▼ description **string**

Description is a human-readable description of the result

▼ name **string** required

Name the given name

▼ properties **object**

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

▼ type **string**

The possible types are 'string', 'array', and 'object', with 'string' as the default.

▼ 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.

▼ allowPrivilegeEscalation `boolean`

AllowPrivilegeEscalation controls whether a process can gain more privileges than its parent process. This bool directly controls if the `no_new_privs` 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.

▼ appArmorProfile `object`

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.

▼ localhostProfile `string`

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".

▼ **type** `string` required

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.

▼ **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.

▼ **add** `[]string`

Added capabilities

▼ **drop** `[]string`

Removed capabilities

▼ **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 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 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.

▼ **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.

▼ **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.

▼ **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** required

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.

▼ windowsOptions **object**

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.

▼ 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.

▼ 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 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.

▼ volumeMounts `[]object`

VolumeMount describes a mounting of a Volume within a container.

▼ mountPath `string` required

Path within the container at which the volume should be mounted. Must not contain '..'.

▼ mountPropagation `string`

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` required

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.

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.

▼ subPath `string`

Path within the volume from which the container's volume should be mounted. Defaults to "" (volume's root).

▼ subPathExpr `string`

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.

▼ `workingDir` `string`

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.

VerificationPolicy [tekton.dev/v1alpha1]

tekton.dev group

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.

v1alpha1 version

▼ spec object required

Spec holds the desired state of the VerificationPolicy.

▼ authorities []object required

The Authority block defines the keys for validating signatures.

▼ key object

Key contains the public key to validate the resource.

▼ data string

Data contains the inline public key.

▼ hashAlgorithm string

HashAlgorithm always defaults to sha256 if the algorithm hasn't been explicitly set

▼ kms **string**

KMS contains the KMS url of the public key Supported formats differ based on the KMS system used. One example of a KMS url could be:

gcpkms://projects/[PROJECT]/locations/[LOCATION]/keyRings/[KEYRING]/cryptoKeys/[KEY]/cryptoKeyVersions/[KEY_VERSION] For more examples please refer https://docs.sigstore.dev/cosign/kms_support.

Note that the KMS is not supported yet.

▼ secretRef **object**

SecretRef sets a reference to a secret with the key.

▼ 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.

▼ name **string** **required**

Name is the name for this authority.

▼ mode **string**

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

▼ resources []object required

ResourcePattern defines the pattern of the resource source

▼ pattern string required

Pattern defines a resource pattern. Regex is created to filter resources based on

Pattern 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/>*,



ResolutionRequest

[resolution.tekton.dev/v1beta1]

`resolution.tekton.dev``group`

ResolutionRequest is an object for requesting the content of a Tekton resource like a pipeline.yaml.

`v1alpha1``version`

▼ spec `object`

Spec holds the information for the request part of the resource request.

▼ params `object`

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.

▼ status `object`

Status communicates the state of the request and, ultimately, the content of the resolved resource.

▼ 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 []object

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> ↗

▼ 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.

▼ status string required

Status of the condition, one of True, False, Unknown.

▼ type string required

Type of condition.

▼ data **string** required

Data is a string representation of the resolved content of the requested resource in-lined into the ResolutionRequest object.

▼ observedGeneration **integer**

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ refSource required

RefSource is the source reference of the remote data that records where the remote file came from including the url, digest and the endpoint.

ResolutionRequest is an object for requesting the content of a Tekton resource like a pipeline.yaml.

v1beta1version**▼ spec** **object**

Spec holds the information for the request part of the resource request.

▼ params **[]object**

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

▼ name **string** required**▼ value** required

▼ url **string**

URL is the runtime url passed to the resolver to help it figure out how to resolve the resource being requested. This is currently at an ALPHA stability level and subject to alpha API compatibility policies.

▼ status **object**

Status communicates the state of the request and, ultimately, the content of the resolved resource.

▼ 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 **[]object**

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> ↗

▼ 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.

▼ **status** `string` `required`

Status of the condition, one of True, False, Unknown.

▼ **type** `string` `required`

Type of condition.

▼ **data** `string` `required`

Data is a string representation of the resolved content of the requested resource in-lined into the ResolutionRequest object.

▼ **observedGeneration** `integer`

ObservedGeneration is the 'Generation' of the Service that was last processed by the controller.

▼ **refSource** `required`

RefSource is the source reference of the remote data that records the url, digest and the endpoint.

▼ **source** `required`

Deprecated: Use RefSource instead