Backup APIs

Backup [velero.io/v1]

Description

Backup is a Velero resource that represents the capture of Kubernetes cluster state at a point in time (API objects and associated volume state).

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	BackupSpec defines the specification for a Velero backup.
`status`	`object`	BackupStatus captures the current status of a Velero backup.

.spec

Description

BackupSpec defines the specification for a Velero backup.

Type

`object`

Property	Type	Description
`csiSnapshotTimeout`	`string`	CSISnapshotTimeout specifies the time used to wait for CSI VolumeSnapshot status turns to ReadyToUse during creation, before returning error as timeout. The default value is 10 minute

Property	Type	Description
`datamover`	`string`	DataMover specifies the data mover to be used by the backup. If DataMover is "" or "velero", the built-in data mover will be used.
`defaultVolumesToFsBackup`	`boolean`	DefaultVolumesToFsBackup specifies whether pod volume file system backup should be used for all volumes by default.
`defaultVolumesToRestic`	`boolean`	DefaultVolumesToRestic specifies whether restic should be used to take a backup of all pod volumes by default. Deprecated: this field is no longer used and will be removed entirely in future. Use DefaultVolumesToFsBackup instead.
`excludedClusterScopedResources`	`array`	ExcludedClusterScopedResources is a slice of cluster-scoped resource type names to exclude from the backup. If set to "*", all cluster-scoped resource types are excluded. The default value is empty.
`excludedNamespaceScopedResources`	`array`	ExcludedNamespaceScopedResources is a slice of namespace-scoped resource type names to exclude from

Property	Type	Description
		the backup. If set to "*", all namespace-scoped resource types are excluded. The default value is empty.
`excludedNamespaces`	`array`	ExcludedNamespaces contains a list of namespaces that are not included in the backup.
`excludedResources`	`array`	ExcludedResources is a slice of resource names that are not included in the backup.
`hooks`	`object`	Hooks represent custom behaviors that should be executed at different phases of the backup.
`includeClusterResources`	`boolean`	IncludeClusterResources specifies whether cluster-scoped resources should be included for consideration in the backup.

Property	Type	Description
`includedClusterScopedResources`	`array`	IncludedClusterScopedResources is a slice of cluster-scoped resource type names to include in the backup. If set t "*", all cluster-scoped resource types are included. The default value is empty, which means only related cluster-scoped resources are included.
`includedNamespaceScopedResources`	`array`	IncludedNamespaceScopedResources is a slice of namespace-scoped resource type names to include in the backup. The default value is "*".
`includedNamespaces`	`array`	IncludedNamespaces is a slice of namespace names to include objects from. If empty, all namespaces are included.
`includedResources`	`array`	IncludedResources is a slice of resource names to include in the backup. If empty, all resources are included.
`itemOperationTimeout`	`string`	ItemOperationTimeout specifies the time used to wait for asynchronous BackupItemAction operations The default value is 4 hour.

Property	Type	Description
`labelSelector`	`object`	LabelSelector is a metav1.LabelSelector to filter with when adding individual objects to the backup. If empty or nil, all objects are included. Optional.
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`orLabelSelectors`	`array`	OrLabelSelectors is list of metav1.LabelSelector to filter with when adding individual objects to the backup. If multiple provided they will be joined by the OR operator. LabelSelector as well as OrLabelSelectors cannot co-exist in backup request, only one of them can be used.
`orderedResources`	`object`	OrderedResources specifies the backup order of resources of specific Kind. The map key is the resource name and value is a list of object names separated by commas. Each resource name has format "namespace/objectname". For cluster resources, simply use "objectname".

Property	Type	Description
`resourcePolicy`	`object`	ResourcePolicy specifies the referenced resource policies that backup should follow
`snapshotMoveData`	`boolean`	SnapshotMoveData specifies whether snapshot data should be moved
`snapshotVolumes`	`boolean`	SnapshotVolumes specifies whether to take snapshots of any PV's referenced in the set of objects included in the Backup.
`storageLocation`	`string`	StorageLocation is a string containing the name of a BackupStorageLocation where the backup should be stored.
`ttl`	`string`	TTL is a time.Duration-parseable string describing how long the Backup should be retained for.
`uploaderConfig`	`object`	UploaderConfig specifies the configuration for the uploader.
`volumeSnapshotLocations`	`array`	VolumeSnapshotLocations is a list containing names of

Property	Type	Description
		VolumeSnapshotLocations associated with this backup.

.spec.excludedClusterScopedResources

Description

ExcludedClusterScopedResources is a slice of cluster-scoped resource type names to exclude from the backup. If set to "*", all cluster-scoped resource types are excluded. The default value is empty.

Type

`array`

.spec.excludedClusterScopedResources[]

Type

`string`

.spec.excludedNamespaceScopedResources

Description

ExcludedNamespaceScopedResources is a slice of namespace-scoped resource type names to exclude from the backup. If set to "*", all namespace-scoped resource types are excluded. The default value is empty.

Type

`array`

.spec.excludedNamespaceScopedResources[]

Type

`string`

.spec.excludedNamespaces

Description

ExcludedNamespaces contains a list of namespaces that are not included in the backup.

Type

`array`

.spec.excludedNamespaces[]

Type

`string`

.spec.excludedResources

Description

ExcludedResources is a slice of resource names that are not included in the backup.

Type

`array`

.spec.excludedResources[]

Type

`string`

.spec.hooks

Description

Hooks represent custom behaviors that should be executed at different phases of the backup.

Type

`object`

Property	Type	Description
`resources`	`array`	Resources are hooks that should be executed when backing up individual instances of a resource.

.spec.hooks.resources

Description

Resources are hooks that should be executed when backing up individual instances of a resource.

Type

`array`

.spec.hooks.resources[]

Description

BackupResourceHookSpec defines one or more BackupResourceHooks that should be executed based on the rules defined for namespaces, resources, and label selector.

Type

`object`

Required

`name`

Property	Type	Description
`excludedNamespaces`	`array`	ExcludedNamespaces specifies the namespaces to which this hook spec does not apply.

Property	Type	Description
`excludedResources`	`array`	ExcludedResources specifies the resources to which this hook spec does not apply.
`includedNamespaces`	`array`	IncludedNamespaces specifies the namespaces to which this hook spec applies. If empty, it applies to all namespaces.
`includedResources`	`array`	IncludedResources specifies the resources to which this hook spec applies. If empty, it applies to all resources.
`labelSelector`	`object`	LabelSelector, if specified, filters the resources to which this hook spec applies.
`name`	`string`	Name is the name of this hook.
`post`	`array`	PostHooks is a list of BackupResourceHooks to execute after storing the item in the backup. These are executed after all "additional items" from item actions are processed.
`pre`	`array`	PreHooks is a list of BackupResourceHooks to execute prior to storing the item in the backup. These are executed before any "additional items" from item actions are processed.

.spec.hooks.resources[].excludedNamespaces

Description

ExcludedNamespaces specifies the namespaces to which this hook spec does not apply.

Type

`array`

.spec.hooks.resources[].excludedNamespaces[]

Type

`string`

.spec.hooks.resources[].excludedResources

Description

ExcludedResources specifies the resources to which this hook spec does not apply.

Type

`array`

.spec.hooks.resources[].excludedResources[]

Type

`string`

.spec.hooks.resources[].includedNamespaces

Description

IncludedNamespaces specifies the namespaces to which this hook spec applies. If empty, it applies to all namespaces.

Type

`array`

.spec.hooks.resources[].includedNamespaces[]

Type

`string`

.spec.hooks.resources[].includedResources

Description

IncludedResources specifies the resources to which this hook spec applies. If empty, it applies to all resources.

Type

`array`

.spec.hooks.resources[].includedResources[]

Type

`string`

.spec.hooks.resources[].labelSelector

Description

LabelSelector, if specified, filters the resources to which this hook spec applies.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.

Property	Type	Description
`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.

.spec.hooks.resources[].labelSelector.matchExpressions

Description

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

Type

`array`

.spec.hooks.resources[].labelSelector.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.

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

.spec.hooks.resources[].labelSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.hooks.resources[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.hooks.resources[].labelSelector.matchLabels

Description

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

Type

`object`

.spec.hooks.resources[].post

Description

PostHooks is a list of BackupResourceHooks to execute after storing the item in the backup. These are executed after all "additional items" from item actions are processed.

Type

`array`

.spec.hooks.resources[].post[]

Description

BackupResourceHook defines a hook for a resource.

Type

`object`

Required

`exec`

Property	Type	Description
`exec`	`object`	Exec defines an exec hook.

.spec.hooks.resources[].post[].exec

Description

Exec defines an exec hook.

Type

`object`

Required

`command`

Property	Type	Description
`command`	`array`	Command is the command and arguments to execute.
`container`	`string`	Container is the container in the pod where the command should be executed. If not specified, the pod's first container is used.
`onError`	`string`	OnError specifies how Velero should behave if it encounters an error executing this hook.
`timeout`	`string`	Timeout defines the maximum amount of time Velero should wait for the hook to complete before considering the execution a failure.

.spec.hooks.resources[].post[].exec.command

Description

Command is the command and arguments to execute.

Type

`array`

.spec.hooks.resources[].post[].exec.command[]

Type

`string`

.spec.hooks.resources[].pre

Description

PreHooks is a list of BackupResourceHooks to execute prior to storing the item in the backup. These are executed before any "additional items" from item actions are processed.

Type

`array`

.spec.hooks.resources[].pre[]

Description

BackupResourceHook defines a hook for a resource.

Type

`object`

Required

`exec`

Property	Type	Description
`exec`	`object`	Exec defines an exec hook.

.spec.hooks.resources[].pre[].exec

Description

Exec defines an exec hook.

Type

`object`

Required

`command`

Property	Type	Description
`command`	`array`	Command is the command and arguments to execute.
`container`	`string`	Container is the container in the pod where the command should be executed. If not specified, the pod's first container is used.
`onError`	`string`	OnError specifies how Velero should behave if it encounters an error executing this hook.
`timeout`	`string`	Timeout defines the maximum amount of time Velero should wait for the hook to complete before considering the execution a failure.

.spec.hooks.resources[].pre[].exec.command

Description

Command is the command and arguments to execute.

Type

`array`

.spec.hooks.resources[].pre[].exec.command[]

Type

`string`

.spec.includedClusterScopedResources

Description

IncludedClusterScopedResources is a slice of cluster-scoped resource type names to include in the backup. If set to "*", all cluster-scoped resource types are included. The default value is empty, which means only related cluster-scoped resources are included.

Type

`array`

.spec.includedClusterScopedResources[]

Type

`string`

.spec.includedNamespaceScopedResources

Description

IncludedNamespaceScopedResources is a slice of namespace-scoped resource type names to include in the backup. The default value is "*".

Type

`array`

.spec.includedNamespaceScopedResources[]

Type

`string`

.spec.includedNamespaces

Description

IncludedNamespaces is a slice of namespace names to include objects from. If empty, all namespaces are included.

Type

`array`

.spec.includedNamespaces[]

Type

`string`

.spec.includedResources

Description

IncludedResources is a slice of resource names to include in the backup. If empty, all resources are included.

Type

`array`

.spec.includedResources[]

Type

`string`

.spec.labelSelector

Description

LabelSelector is a metav1.LabelSelector to filter with when adding individual objects to the backup. If empty or nil, all objects are included. Optional.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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.

.spec.labelSelector.matchExpressions

Description

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

Type

`array`

.spec.labelSelector.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.labelSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.labelSelector.matchLabels

Description

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

Type

`object`

.spec.orLabelSelectors

Description

OrLabelSelectors is list of metav1.LabelSelector to filter with when adding individual objects to the backup. If multiple provided they will be joined by the OR operator. LabelSelector as well as OrLabelSelectors cannot co-exist in backup request, only one of them can be used.

Type

`array`

.spec.orLabelSelectors[]

Description

A label selector is a label query over a set of resources. The result of matchLabels and matchExpressions are ANDed. An empty label selector matches all objects. A null label selector matches no objects.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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

Property	Type	Description
		"key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.orLabelSelectors[].matchExpressions

Description

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

Type

`array`

.spec.orLabelSelectors[].matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

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

.spec.orLabelSelectors[].matchExpressions[].values

Description

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

Type

`array`

.spec.orLabelSelectors[].matchExpressions[].values[]

Type

`string`

.spec.orLabelSelectors[].matchLabels

Description

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

Type

`object`

.spec.orderedResources

Description

OrderedResources specifies the backup order of resources of specific Kind. The map key is the resource name and value is a list of object names separated by commas. Each resource name has format "namespace/objectname". For cluster resources, simply use "objectname".

Type

`object`

.spec.resourcePolicy

Description

ResourcePolicy specifies the referenced resource policies that backup should follow

Type

`object`

Required

`kind` `name`

Property	Type	Description
`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`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.uploaderConfig

Description

UploaderConfig specifies the configuration for the uploader.

Type

`object`

Property	Type	Description
`parallelFilesUpload`	`integer`	ParallelFilesUpload is the number of files parallel uploads to perform when using the uploader.

.spec.volumeSnapshotLocations

Description

VolumeSnapshotLocations is a list containing names of VolumeSnapshotLocations associated with this backup.

Type

`array`

.spec.volumeSnapshotLocations[]

Type

`string`

.status

Description

BackupStatus captures the current status of a Velero backup.

Type

`object`

Property	Type	Description
`backupItemOperationsAttempted`	`integer`	BackupItemOperationsAttempted is the total number of attempted async BackupItemAction operations for this backup.
`backupItemOperationsCompleted`	`integer`	BackupItemOperationsCompleted is the total number of successfully completed async BackupItemAction operations for this backup.
`backupItemOperationsFailed`	`integer`	BackupItemOperationsFailed is the total number of async BackupItemAction operations for this backup which ended with an error.
`completionTimestamp`	`string`	CompletionTimestamp records the time a backup was completed. Completion time is recorded even on failed backups. Completion time is recorded before uploading the backup object. The server's time is used for CompletionTimestamps
`csiVolumeSnapshotsAttempted`	`integer`	CSIVolumeSnapshotsAttempted is the total number of attempted CSI VolumeSnapshots for this backup.

Property	Type	Description
`csiVolumeSnapshotsCompleted`	`integer`	CSIVolumeSnapshotsCompleted is the total number of successfully completed CSI VolumeSnapshots for this backup.
`errors`	`integer`	Errors is a count of all error messages that were generated during execution of the backup. The actual errors are in the backup's log file in object storage.
`expiration`	`string`	Expiration is when this Backup is eligible for garbage-collection.
`failureReason`	`string`	FailureReason is an error that caused the entire backup to fail.
`formatVersion`	`string`	FormatVersion is the backup format version, including major, minor, and patch version.
`hookStatus`	`object`	HookStatus contains information about the status of the hooks.

Property	Type	Description
`phase`	`string`	Phase is the current state of the Backup.
`progress`	`object`	Progress contains information about the backup's execution progress. Note that this information is best-effort only -- if Velero fails to update it during a backup for any reason, it may be inaccurate/stale.
`startTimestamp`	`string`	StartTimestamp records the time a backup was started. Separate from CreationTimestamp, since that value changes on restores. The server's time is used for StartTimestamps
`validationErrors`	`array`	ValidationErrors is a slice of all validation errors (if applicable).
`version`	`integer`	Version is the backup format major version. Deprecated: Please see FormatVersion
`volumeSnapshotsAttempted`	`integer`	VolumeSnapshotsAttempted is the total number of attempted volume snapshots for this backup.

Property	Type	Description
`volumeSnapshotsCompleted`	`integer`	VolumeSnapshotsCompleted is the total number of successfully completed volume snapshots for this backup.
`warnings`	`integer`	Warnings is a count of all warning messages that were generated during execution of the backup. The actual warnings are in the backup's log file in object storage.

.status.hookStatus

Description

HookStatus contains information about the status of the hooks.

Type

`object`

Property	Type	Description
`hooksAttempted`	`integer`	HooksAttempted is the total number of attempted hooks Specifically, HooksAttempted represents the number of hooks that failed to execute and the number of hooks that executed successfully.
`hooksFailed`	`integer`	HooksFailed is the total number of hooks which ended with an error

.status.progress

Description

Progress contains information about the backup's execution progress. Note that this information is best-effort only -- if Velero fails to update it during a backup for any reason, it may be inaccurate/stale.

Type

`object`

Property	Type	Description
`itemsBackedUp`	`integer`	ItemsBackedUp is the number of items that have actually been written to the backup tarball so far.
`totalItems`	`integer`	TotalItems is the total number of items to be backed up. This number may change throughout the execution of the backup due to plugins that return additional related items to back up, the velero.io/exclude-from-backup label, and various other filters that happen as items are processed.

.status.validationErrors

Description

ValidationErrors is a slice of all validation errors (if applicable).

Type

`array`

.status.validationErrors[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/backups`
  - `DELETE`: delete collection of Backup
  - `GET`: list objects of kind Backup
  - `POST`: create a new Backup
- `/apis/velero.io/v1/namespaces/{namespace}/backups/{name}`
  - `DELETE`: delete the specified Backup
  - `GET`: read the specified Backup
  - `PATCH`: partially update the specified Backup
  - `PUT`: replace the specified Backup
- `/apis/velero.io/v1/namespaces/{namespace}/backups/{name}/status`
  - `GET`: read status of the specified Backup
  - `PATCH`: partially update status of the specified Backup
  - `PUT`: replace status of the specified Backup

/apis/velero.io/v1/namespaces/{namespace}/backups

HTTP method

`DELETE`

Description

delete collection of Backup

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind Backup

HTTP responses

HTTP code	Response body
200 - OK	`BackupList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new Backup

Query parameters

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

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

Body parameters

Parameter	Type	Description
`body`	`Backup` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
201 - Created	`Backup` schema
202 - Accepted	`Backup` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/backups/{name}

HTTP method

`DELETE`

Description

delete the specified Backup

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified Backup

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified Backup

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified Backup

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`Backup` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
201 - Created	`Backup` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/backups/{name}/status

HTTP method

`GET`

Description

read status of the specified Backup

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified Backup

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

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

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified Backup

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`Backup` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Backup` schema
201 - Created	`Backup` schema

HTTP code	Response body
401 - Unauthorized	Empty

BackupRepository [velero.io/v1]

Type

`object`

Specification

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

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	BackupRepositorySpec is the specification for a BackupRepository.
`status`	`object`	BackupRepositoryStatus is the current status of a BackupRepository.

.spec

Description

BackupRepositorySpec is the specification for a BackupRepository.

Type

`object`

Required

`backupStorageLocation` `maintenanceFrequency` `resticIdentifier` `volumeNamespace`

Property	Type	Description
`backupStorageLocation`	`string`	BackupStorageLocation is the name of the BackupStorageLocation that should contain this repository.

Property	Type	Description
`maintenanceFrequency`	`string`	MaintenanceFrequency is how often maintenance should be run.
`repositoryConfig`	`object`	RepositoryConfig is for repository-specific configuration fields.
`repositoryType`	`string`	RepositoryType indicates the type of the backend repository
`resticIdentifier`	`string`	ResticIdentifier is the full restic-compatible string for identifying this repository.
`volumeNamespace`	`string`	VolumeNamespace is the namespace this backup repository contains pod volume backups for.

.spec.repositoryConfig

Description

RepositoryConfig is for repository-specific configuration fields.

Type

`object`

.status

Description

BackupRepositoryStatus is the current status of a BackupRepository.

Type

`object`

Property	Type	Description
`lastMaintenanceTime`	`string`	LastMaintenanceTime is the last time maintenance was run.
`message`	`string`	Message is a message about the current status of the BackupRepository.
`phase`	`string`	Phase is the current state of the BackupRepository.

API Endpoints

The following API endpoints are available:

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

- `/apis/velero.io/v1/namespaces/{namespace}/backuprepositories/{name}/status`
  - `GET`: read status of the specified BackupRepository
  - `PATCH`: partially update status of the specified BackupRepository
  - `PUT`: replace status of the specified BackupRepository

/apis/velero.io/v1/namespaces/{namespace}/backuprepositories

HTTP method

`DELETE`

Description

delete collection of BackupRepository

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind BackupRepository

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepositoryList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new BackupRepository

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`BackupRepository` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
201 - Created	`BackupRepository` schema
202 - Accepted	`BackupRepository` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/backuprepositories/{name}

HTTP method

`DELETE`

Description

delete the specified BackupRepository

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified BackupRepository

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified BackupRepository

Query parameters

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

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified BackupRepository

Query parameters

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

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BackupRepository` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
201 - Created	`BackupRepository` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/backuprepositories/{name}/status

HTTP method

`GET`

Description

read status of the specified BackupRepository

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified BackupRepository

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified BackupRepository

Query parameters

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

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

Body parameters

Parameter	Type	Description
`body`	`BackupRepository` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BackupRepository` schema
201 - Created	`BackupRepository` schema
401 - Unauthorized	Empty

BackupStorageLocation [velero.io/v1]

Description

BackupStorageLocation is a location where Velero stores backup objects

Type

`object`

Specification

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

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

.spec

Description

BackupStorageLocationSpec defines the desired state of a Velero BackupStorageLocation

Type

`object`

Required

`objectStorage` `provider`

Property	Type	Description
`accessMode`	`string`	AccessMode defines the permissions for the backup storage location.
`backupSyncPeriod`	`string`	BackupSyncPeriod defines how frequently to sync backup API objects from object storage. A value of 0 disables sync.
`config`	`object`	Config is for provider-specific configuration fields.
`credential`	`object`	Credential contains the credential information intended to be used with this location
`default`	`boolean`	Default indicates this location is the default backup storage location.
`objectStorage`	`object`	ObjectStorageLocation specifies the settings necessary to connect to a provider's object storage.
`provider`	`string`	Provider is the provider of the backup storage.

Property	Type	Description
`validationFrequency`	`string`	ValidationFrequency defines how frequently to validate the corresponding object storage. A value of 0 disables validation.

.spec.config

Description

Config is for provider-specific configuration fields.

Type

`object`

.spec.credential

Description

Credential contains the credential information intended to be used with this location

Type

`object`

Required

`key`

Property	Type	Description
`key`	`string`	The key of the secret to select from. Must be a valid secret key.
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-

Property	Type	Description
		objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?
`optional`	`boolean`	Specify whether the Secret or its key must be defined

.spec.objectStorage

Description

ObjectStorageLocation specifies the settings necessary to connect to a provider's object storage.

Type

`object`

Required

`bucket`

Property	Type	Description
`bucket`	`string`	Bucket is the bucket to use for object storage.
`caCert`	`string`	CACert defines a CA bundle to use when verifying TLS connections to the provider.
`prefix`	`string`	Prefix is the path inside a bucket to use for Velero storage. Optional.

.status

Description

BackupStorageLocationStatus defines the observed state of BackupStorageLocation

Type

`object`

Property	Type	Description
`accessMode`	`string`	AccessMode is an unused field. Deprecated: there is now an AccessMode field on the Spec and this field will be removed entirely as of v2.0.
`lastSyncedRevision`	`string`	LastSyncedRevision is the value of the `metadata/revision` file in the backup storage location the last time the BSL's contents were synced into the cluster. Deprecated: this field is no longer updated or used for detecting changes to the location's contents and will be removed entirely in v2.0.
`lastSyncedTime`	`string`	LastSyncedTime is the last time the contents of the location were synced into the cluster.
`lastValidationTime`	`string`	LastValidationTime is the last time the backup store location was validated the cluster.

Property	Type	Description
`message`	`string`	Message is a message about the backup storage location's status.
`phase`	`string`	Phase is the current state of the BackupStorageLocation.

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/backupstoragelocations`
  - `DELETE`: delete collection of BackupStorageLocation
  - `GET`: list objects of kind BackupStorageLocation
  - `POST`: create a new BackupStorageLocation
- `/apis/velero.io/v1/namespaces/{namespace}/backupstoragelocations/{name}`
  - `DELETE`: delete the specified BackupStorageLocation
  - `GET`: read the specified BackupStorageLocation
  - `PATCH`: partially update the specified BackupStorageLocation
  - `PUT`: replace the specified BackupStorageLocation
- `/apis/velero.io/v1/namespaces/{namespace}/backupstoragelocations/{name}/status`
  - `GET`: read status of the specified BackupStorageLocation
  - `PATCH`: partially update status of the specified BackupStorageLocation
  - `PUT`: replace status of the specified BackupStorageLocation

/apis/velero.io/v1/namespaces/{namespace}/backupstoragelocations

HTTP method

`DELETE`

Description

delete collection of BackupStorageLocation

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind BackupStorageLocation

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocationList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new BackupStorageLocation

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`BackupStorageLocation` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema
201 - Created	`BackupStorageLocation` schema

HTTP code	Response body
202 - Accepted	`BackupStorageLocation` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/backupstoragelocations/{name}

HTTP method

`DELETE`

Description

delete the specified BackupStorageLocation

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified BackupStorageLocation

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified BackupStorageLocation

Query parameters

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

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified BackupStorageLocation

Query parameters

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

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`BackupStorageLocation` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema
201 - Created	`BackupStorageLocation` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/backupstoragelocations/{name}/status

HTTP method

`GET`

Description

read status of the specified BackupStorageLocation

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified BackupStorageLocation

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified BackupStorageLocation

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`BackupStorageLocation` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`BackupStorageLocation` schema
201 - Created	`BackupStorageLocation` schema
401 - Unauthorized	Empty

DataDownload [velero.io/v2alpha1]

Description

DataDownload acts as the protocol between data mover plugins and data mover controller for the datamover restore operation

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataDownloadSpec is the specification for a DataDownload.
`status`	`object`	DataDownloadStatus is the current status of a DataDownload.

.spec

Description

DataDownloadSpec is the specification for a DataDownload.

Type

`object`

Required

`backupStorageLocation` `operationTimeout` `snapshotID` `sourceNamespace` `targetVolume`

Property	Type	Description
`backupStorageLocation`	`string`	BackupStorageLocation is the name of the backup storage location where the backup repository is stored.
`cancel`	`boolean`	Cancel indicates request to cancel the ongoing DataDownload. It can be set when the DataDownload is in InProgress phase
`dataMoverConfig`	`object`	DataMoverConfig is for data-mover-specific configuration fields.
`datamover`	`string`	DataMover specifies the data mover to be used by the backup. If DataMover is "" or "velero", the built-in data mover will be used.
`operationTimeout`	`string`	OperationTimeout specifies the time used to wait internal operations, before returning error as timeout.
`snapshotID`	`string`	SnapshotID is the ID of the Velero backup snapshot to be restored from.

Property	Type	Description
`sourceNamespace`	`string`	SourceNamespace is the original namespace where the volume is backed up from. It may be different from SourcePVC's namespace if namespace is remapped during restore.
`targetVolume`	`object`	TargetVolume is the information of the target PVC and PV.

.spec.dataMoverConfig

Description

DataMoverConfig is for data-mover-specific configuration fields.

Type

`object`

.spec.targetVolume

Description

TargetVolume is the information of the target PVC and PV.

Type

`object`

Required

`namespace` `pv` `pvc`

Property	Type	Description
`namespace`	`string`	Namespace is the target namespace

Property	Type	Description
`pv`	`string`	PV is the name of the target PV that is created by Velero restore
`pvc`	`string`	PVC is the name of the target PVC that is created by Velero restore

.status

Description

DataDownloadStatus is the current status of a DataDownload.

Type

`object`

Property	Type	Description
`completionTimestamp`	`string`	CompletionTimestamp records the time a restore was completed. Completion time is recorded even on failed restores. The server's time is used for CompletionTimestamps
`message`	`string`	Message is a message about the DataDownload's status.
`node`	`string`	Node is name of the node where the DataDownload is processed.

Property	Type	Description
`phase`	`string`	Phase is the current state of the DataDownload.
`progress`	`object`	Progress holds the total number of bytes of the snapshot and the current number of restored bytes. This can be used to display progress information about the restore operation.
`startTimestamp`	`string`	StartTimestamp records the time a restore was started. The server's time is used for StartTimestamps

.status.progress

Description

Progress holds the total number of bytes of the snapshot and the current number of restored bytes. This can be used to display progress information about the restore operation.

Type

`object`

Property	Type	Description
`bytesDone`	`integer`	
`totalBytes`	`integer`	

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v2alpha1/namespaces/{namespace}/datadownloads`
  - `DELETE`: delete collection of DataDownload
  - `GET`: list objects of kind DataDownload
  - `POST`: create a new DataDownload
- `/apis/velero.io/v2alpha1/namespaces/{namespace}/datadownloads/{name}`
  - `DELETE`: delete the specified DataDownload
  - `GET`: read the specified DataDownload
  - `PATCH`: partially update the specified DataDownload
  - `PUT`: replace the specified DataDownload
- `/apis/velero.io/v2alpha1/namespaces/{namespace}/datadownloads/{name}/status`
  - `GET`: read status of the specified DataDownload
  - `PATCH`: partially update status of the specified DataDownload
  - `PUT`: replace status of the specified DataDownload

/apis/velero.io/v2alpha1/namespaces/{namespace}/datadownloads

HTTP method

`DELETE`

Description

delete collection of DataDownload

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind DataDownload

HTTP responses

HTTP code	Response body
200 - OK	`DataDownloadList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DataDownload

Query parameters

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

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataDownload` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema
201 - Created	`DataDownload` schema
202 - Accepted	`DataDownload` schema
401 - Unauthorized	Empty

/apis/velero.io/v2alpha1/namespaces/{namespace}/datadownloads/{name}

HTTP method

`DELETE`

Description

delete the specified DataDownload

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified DataDownload

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DataDownload

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DataDownload

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`DataDownload` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema

HTTP code	Response body
201 - Created	`DataDownload` schema
401 - Unauthorized	Empty

/apis/velero.io/v2alpha1/namespaces/{namespace}/datadownloads/{name}/status

HTTP method

`GET`

Description

read status of the specified DataDownload

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DataDownload

Query parameters

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

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

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DataDownload

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

Body parameters

Parameter	Type	Description
`body`	`DataDownload` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataDownload` schema
201 - Created	`DataDownload` schema
401 - Unauthorized	Empty

DataUpload [velero.io/v2alpha1]

Description

DataUpload acts as the protocol between data mover plugins and data mover controller for the datamover backup operation

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DataUploadSpec is the specification for a DataUpload.
`status`	`object`	DataUploadStatus is the current status of a DataUpload.

.spec

Description

DataUploadSpec is the specification for a DataUpload.

Type

`object`

Required

`backupStorageLocation` `operationTimeout` `snapshotType` `sourceNamespace` `sourcePVC`

Property	Type	Description
`backupStorageLocation`	`string`	BackupStorageLocation is the name of the backup storage location where the backup repository is stored.

Property	Type	Description
`cancel`	`boolean`	Cancel indicates request to cancel the ongoing DataUpload. It can be set when the DataUpload is in InProgress phase
`csiSnapshot`	`object`	If SnapshotType is CSI, CSISnapshot provides the information of the CSI snapshot.
`dataMoverConfig`	`object`	DataMoverConfig is for data-mover-specific configuration fields.
`datamover`	`string`	DataMover specifies the data mover to be used by the backup. If DataMover is "" or "velero", the built-in data mover will be used.
`operationTimeout`	`string`	OperationTimeout specifies the time used to wait internal operations, before returning error as timeout.
`snapshotType`	`string`	SnapshotType is the type of the snapshot to be backed up.
`sourceNamespace`	`string`	SourceNamespace is the original namespace where the volume is backed up from. It is the same namespace for SourcePVC and CSI namespaced objects.

Property	Type	Description
`sourcePVC`	`string`	SourcePVC is the name of the PVC which the snapshot is taken for.

.spec.csiSnapshot

Description

If SnapshotType is CSI, CSISnapshot provides the information of the CSI snapshot.

Type

`object`

Required

`storageClass` `volumeSnapshot`

Property	Type	Description
`snapshotClass`	`string`	SnapshotClass is the name of the snapshot class that the volume snapshot is created with
`storageClass`	`string`	StorageClass is the name of the storage class of the PVC that the volume snapshot is created from
`volumeSnapshot`	`string`	VolumeSnapshot is the name of the volume snapshot to be backed up

.spec.dataMoverConfig

Description

DataMoverConfig is for data-mover-specific configuration fields.

Type

`object`

.status

Description

DataUploadStatus is the current status of a DataUpload.

Type

`object`

Property	Type	Description
`completionTimestamp`	`string`	CompletionTimestamp records the time a backup was completed. Completion time is recorded even on failed backups. Completion time is recorded before uploading the backup object. The server's time is used for CompletionTimestamps
`dataMoverResult`	`object`	DataMoverResult stores data-mover-specific information as a result of the DataUpload.
`message`	`string`	Message is a message about the DataUpload's status.
`node`	`string`	Node is name of the node where the DataUpload is processed.

Property	Type	Description
`path`	`string`	Path is the full path of the snapshot volume being backed up.
`phase`	`string`	Phase is the current state of the DataUpload.
`progress`	`object`	Progress holds the total number of bytes of the volume and the current number of backed up bytes. This can be used to display progress information about the backup operation.
`snapshotID`	`string`	SnapshotID is the identifier for the snapshot in the backup repository.
`startTimestamp`	`string`	StartTimestamp records the time a backup was started. Separate from CreationTimestamp, since that value changes on restores. The server's time is used for StartTimestamps

.status.dataMoverResult

Description

DataMoverResult stores data-mover-specific information as a result of the DataUpload.

Type

`object`

.status.progress

Description

Progress holds the total number of bytes of the volume and the current number of backed up bytes. This can be used to display progress information about the backup operation.

Type

`object`

Property	Type	Description
`bytesDone`	`integer`	
`totalBytes`	`integer`	

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v2alpha1/namespaces/{namespace}/datauploads`
  - `DELETE`: delete collection of DataUpload
  - `GET`: list objects of kind DataUpload
  - `POST`: create a new DataUpload
- `/apis/velero.io/v2alpha1/namespaces/{namespace}/datauploads/{name}`
  - `DELETE`: delete the specified DataUpload
  - `GET`: read the specified DataUpload
  - `PATCH`: partially update the specified DataUpload
  - `PUT`: replace the specified DataUpload
- `/apis/velero.io/v2alpha1/namespaces/{namespace}/datauploads/{name}/status`
  - `GET`: read status of the specified DataUpload
  - `PATCH`: partially update status of the specified DataUpload
  - `PUT`: replace status of the specified DataUpload

/apis/velero.io/v2alpha1/namespaces/{namespace}/datauploads

HTTP method

`DELETE`

Description

delete collection of DataUpload

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind DataUpload

HTTP responses

HTTP code	Response body
200 - OK	`DataUploadList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DataUpload

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`DataUpload` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema
201 - Created	`DataUpload` schema

HTTP code	Response body
202 - Accepted	`DataUpload` schema
401 - Unauthorized	Empty

/apis/velero.io/v2alpha1/namespaces/{namespace}/datauploads/{name}

HTTP method

`DELETE`

Description

delete the specified DataUpload

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified DataUpload

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DataUpload

Query parameters

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

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DataUpload

Query parameters

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

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DataUpload` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema
201 - Created	`DataUpload` schema
401 - Unauthorized	Empty

/apis/velero.io/v2alpha1/namespaces/{namespace}/datauploads/{name}/status

HTTP method

`GET`

Description

read status of the specified DataUpload

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DataUpload

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DataUpload

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`DataUpload` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DataUpload` schema
201 - Created	`DataUpload` schema
401 - Unauthorized	Empty

DeleteBackupRequest [velero.io/v1]

Description

DeleteBackupRequest is a request to delete one or more backups.

Type

`object`

Specification

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

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DeleteBackupRequestSpec is the specification for which backups to delete.
`status`	`object`	DeleteBackupRequestStatus is the current status of a DeleteBackupRequest.

.spec

Description

DeleteBackupRequestSpec is the specification for which backups to delete.

Type

`object`

Required

`backupName`

Property	Type	Description
`backupName`	`string`	

.status

Description

DeleteBackupRequestStatus is the current status of a DeleteBackupRequest.

Type

`object`

Property	Type	Description
`errors`	`array`	Errors contains any errors that were encountered during the deletion process.
`phase`	`string`	Phase is the current state of the DeleteBackupRequest.

.status.errors

Description

Errors contains any errors that were encountered during the deletion process.

Type

`array`

.status.errors[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/deletebackuprequests`
  - `DELETE`: delete collection of DeleteBackupRequest
  - `GET`: list objects of kind DeleteBackupRequest
  - `POST`: create a new DeleteBackupRequest
- `/apis/velero.io/v1/namespaces/{namespace}/deletebackuprequests/{name}`
  - `DELETE`: delete the specified DeleteBackupRequest
  - `GET`: read the specified DeleteBackupRequest
  - `PATCH`: partially update the specified DeleteBackupRequest
  - `PUT`: replace the specified DeleteBackupRequest
- `/apis/velero.io/v1/namespaces/{namespace}/deletebackuprequests/{name}/status`
  - `GET`: read status of the specified DeleteBackupRequest
  - `PATCH`: partially update status of the specified DeleteBackupRequest
  - `PUT`: replace status of the specified DeleteBackupRequest

/apis/velero.io/v1/namespaces/{namespace}/deletebackuprequests

HTTP method

`DELETE`

Description

delete collection of DeleteBackupRequest

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind DeleteBackupRequest

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequestList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DeleteBackupRequest

Query parameters

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

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

Body parameters

Parameter	Type	Description
`body`	`DeleteBackupRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
201 - Created	`DeleteBackupRequest` schema
202 - Accepted	`DeleteBackupRequest` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/deletebackuprequests/{name}

HTTP method

`DELETE`

Description

delete the specified DeleteBackupRequest

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified DeleteBackupRequest

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DeleteBackupRequest

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DeleteBackupRequest

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`DeleteBackupRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
201 - Created	`DeleteBackupRequest` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/deletebackuprequests/{name}/status

HTTP method

`GET`

Description

read status of the specified DeleteBackupRequest

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DeleteBackupRequest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

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

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DeleteBackupRequest

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`DeleteBackupRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DeleteBackupRequest` schema
201 - Created	`DeleteBackupRequest` schema

HTTP code	Response body
401 - Unauthorized	Empty

DownloadRequest [velero.io/v1]

Description

DownloadRequest is a request to download an artifact from backup object storage, such as a backup log file.

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	DownloadRequestSpec is the specification for a download request.
`status`	`object`	DownloadRequestStatus is the current status of a DownloadRequest.

.spec

Description

DownloadRequestSpec is the specification for a download request.

Type

`object`

Required

`target`

Property	Type	Description
`target`	`object`	Target is what to download (e.g. logs for a backup).

.spec.target

Description

Target is what to download (e.g. logs for a backup).

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	Kind is the type of file to download.
`name`	`string`	Name is the name of the Kubernetes resource with which the file is associated.

.status

Description

DownloadRequestStatus is the current status of a DownloadRequest.

Type

`object`

Property	Type	Description
`downloadURL`	`string`	DownloadURL contains the pre-signed URL for the target file.
`expiration`	`string`	Expiration is when this DownloadRequest expires and can be deleted by the system.

Property	Type	Description
`phase`	`string`	Phase is the current state of the DownloadRequest.

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/downloadrequests`
  - `DELETE`: delete collection of DownloadRequest
  - `GET`: list objects of kind DownloadRequest
  - `POST`: create a new DownloadRequest
- `/apis/velero.io/v1/namespaces/{namespace}/downloadrequests/{name}`
  - `DELETE`: delete the specified DownloadRequest
  - `GET`: read the specified DownloadRequest
  - `PATCH`: partially update the specified DownloadRequest
  - `PUT`: replace the specified DownloadRequest
- `/apis/velero.io/v1/namespaces/{namespace}/downloadrequests/{name}/status`
  - `GET`: read status of the specified DownloadRequest
  - `PATCH`: partially update status of the specified DownloadRequest
  - `PUT`: replace status of the specified DownloadRequest

/apis/velero.io/v1/namespaces/{namespace}/downloadrequests

HTTP method

`DELETE`

Description

delete collection of DownloadRequest

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind DownloadRequest

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequestList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new DownloadRequest

Query parameters

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

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

Body parameters

Parameter	Type	Description
`body`	`DownloadRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
201 - Created	`DownloadRequest` schema
202 - Accepted	`DownloadRequest` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/downloadrequests/{name}

HTTP method

`DELETE`

Description

delete the specified DownloadRequest

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified DownloadRequest

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified DownloadRequest

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified DownloadRequest

Query parameters

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

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

Body parameters

Parameter	Type	Description
`body`	`DownloadRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
201 - Created	`DownloadRequest` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/downloadrequests/{name}/status

HTTP method

`GET`

Description

read status of the specified DownloadRequest

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified DownloadRequest

Query parameters

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

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified DownloadRequest

Query parameters

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

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`DownloadRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`DownloadRequest` schema
201 - Created	`DownloadRequest` schema
401 - Unauthorized	Empty

PodVolumeBackup [velero.io/v1]

Type

`object`

Specification

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

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	PodVolumeBackupSpec is the specification for a PodVolumeBackup.
`status`	`object`	PodVolumeBackupStatus is the current status of a PodVolumeBackup.

.spec

Description

PodVolumeBackupSpec is the specification for a PodVolumeBackup.

Type

`object`

Required

`backupStorageLocation` `node` `pod` `repoIdentifier` `volume`

Property	Type	Description
`backupStorageLocation`	`string`	BackupStorageLocation is the name of the backup storage location where the backup repository is stored.

Property	Type	Description
`node`	`string`	Node is the name of the node that the Pod is running on.
`pod`	`object`	Pod is a reference to the pod containing the volume to be backed up.
`repoIdentifier`	`string`	RepoIdentifier is the backup repository identifier.
`tags`	`object`	Tags are a map of key-value pairs that should be applied to the volume backup as tags.
`uploaderSettings`	`object`	UploaderSettings are a map of key-value pairs that should be applied to the uploader configuration.
`uploaderType`	`string`	UploaderType is the type of the uploader to handle the data transfer.
`volume`	`string`	Volume is the name of the volume within the Pod to be backed up.

.spec.pod

Description

Pod is a reference to the pod containing the volume to be backed up.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object. TODO: this design is not final and this field is subject to change in the future.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.tags

Description

Tags are a map of key-value pairs that should be applied to the volume backup as tags.

Type

`object`

.spec.uploaderSettings

Description

UploaderSettings are a map of key-value pairs that should be applied to the uploader configuration.

Type

`object`

.status

Description

PodVolumeBackupStatus is the current status of a PodVolumeBackup.

Type

`object`

Property	Type	Description
`completionTimestamp`	`string`	CompletionTimestamp records the time a backup was completed. Completion time is recorded even on failed backups. Completion time is recorded before uploading the backup object. The server's time is used for CompletionTimestamps
`message`	`string`	Message is a message about the pod volume backup's status.
`path`	`string`	Path is the full path within the controller pod being backed up.
`phase`	`string`	Phase is the current state of the PodVolumeBackup.

Property	Type	Description
`progress`	`object`	Progress holds the total number of bytes of the volume and the current number of backed up bytes. This can be used to display progress information about the backup operation.
`snapshotID`	`string`	SnapshotID is the identifier for the snapshot of the pod volume.
`startTimestamp`	`string`	StartTimestamp records the time a backup was started. Separate from CreationTimestamp, since that value changes on restores. The server's time is used for StartTimestamps

.status.progress

Description

Progress holds the total number of bytes of the volume and the current number of backed up bytes. This can be used to display progress information about the backup operation.

Type

`object`

Property	Type	Description
`bytesDone`	`integer`	
`totalBytes`	`integer`	

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/podvolumebackups`
  - `DELETE`: delete collection of PodVolumeBackup
  - `GET`: list objects of kind PodVolumeBackup
  - `POST`: create a new PodVolumeBackup
- `/apis/velero.io/v1/namespaces/{namespace}/podvolumebackups/{name}`
  - `DELETE`: delete the specified PodVolumeBackup
  - `GET`: read the specified PodVolumeBackup
  - `PATCH`: partially update the specified PodVolumeBackup
  - `PUT`: replace the specified PodVolumeBackup
- `/apis/velero.io/v1/namespaces/{namespace}/podvolumebackups/{name}/status`
  - `GET`: read status of the specified PodVolumeBackup
  - `PATCH`: partially update status of the specified PodVolumeBackup
  - `PUT`: replace status of the specified PodVolumeBackup

/apis/velero.io/v1/namespaces/{namespace}/podvolumebackups

HTTP method

`DELETE`

Description

delete collection of PodVolumeBackup

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind PodVolumeBackup

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackupList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new PodVolumeBackup

Query parameters

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

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`PodVolumeBackup` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema
201 - Created	`PodVolumeBackup` schema
202 - Accepted	`PodVolumeBackup` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/podvolumebackups/{name}

HTTP method

`DELETE`

Description

delete the specified PodVolumeBackup

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified PodVolumeBackup

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified PodVolumeBackup

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified PodVolumeBackup

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`PodVolumeBackup` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema

HTTP code	Response body
201 - Created	`PodVolumeBackup` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/podvolumebackups/{name}/status

HTTP method

`GET`

Description

read status of the specified PodVolumeBackup

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified PodVolumeBackup

Query parameters

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

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

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified PodVolumeBackup

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

Body parameters

Parameter	Type	Description
`body`	`PodVolumeBackup` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeBackup` schema
201 - Created	`PodVolumeBackup` schema
401 - Unauthorized	Empty

PodVolumeRestore [velero.io/v1]

Type

`object`

Specification

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

Property	Type	Description
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	PodVolumeRestoreSpec is the specification for a PodVolumeRestore.
`status`	`object`	PodVolumeRestoreStatus is the current status of a PodVolumeRestore.

.spec

Description

PodVolumeRestoreSpec is the specification for a PodVolumeRestore.

Type

`object`

Required

`backupStorageLocation` `pod` `repoIdentifier` `snapshotID` `sourceNamespace` `volume`

Property	Type	Description
`backupStorageLocation`	`string`	BackupStorageLocation is the name of the backup storage location where the backup repository is stored.

Property	Type	Description
`pod`	`object`	Pod is a reference to the pod containing the volume to be restored.
`repoIdentifier`	`string`	RepoIdentifier is the backup repository identifier.
`snapshotID`	`string`	SnapshotID is the ID of the volume snapshot to be restored.
`sourceNamespace`	`string`	SourceNamespace is the original namespace for namaspace mapping.
`uploaderSettings`	`object`	UploaderSettings are a map of key-value pairs that should be applied to the uploader configuration.
`uploaderType`	`string`	UploaderType is the type of the uploader to handle the data transfer.
`volume`	`string`	Volume is the name of the volume within the Pod to be restored.

.spec.pod

Description

Pod is a reference to the pod containing the volume to be restored.

Type

`object`

Property	Type	Description
`apiVersion`	`string`	API version of the referent.
`fieldPath`	`string`	If referring to a piece of an object instead of an entire object, this string should contain a valid JSON/Go field access statement, such as desiredState.manifest.containers[2]. For example, if the object reference is to a container within a pod, this would take on a value like: "spec.containers{name}" (where "name" refers to the name of the container that triggered the event) or if no container name is specified "spec.containers[2]" (container with index 2 in this pod). This syntax is chosen only to have some well-defined way of referencing a part of an object. TODO: this design is not final and this field is subject to change in the future.
`kind`	`string`	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗

Property	Type	Description
`namespace`	`string`	Namespace of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/namespaces/ ↗
`resourceVersion`	`string`	Specific resourceVersion to which this reference is made, if any. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#concurrency-control-and-consistency ↗
`uid`	`string`	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids ↗

.spec.uploaderSettings

Description

UploaderSettings are a map of key-value pairs that should be applied to the uploader configuration.

Type

`object`

.status

Description

PodVolumeRestoreStatus is the current status of a PodVolumeRestore.

Type

`object`

Property	Type	Description
`completionTimestamp`	`string`	CompletionTimestamp records the time a restore was completed. Completion time is recorded even on failed restores. The server's time is used for CompletionTimestamps
`message`	`string`	Message is a message about the pod volume restore's status.
`phase`	`string`	Phase is the current state of the PodVolumeRestore.
`progress`	`object`	Progress holds the total number of bytes of the snapshot and the current number of restored bytes. This can be used to display progress information about the restore operation.
`startTimestamp`	`string`	StartTimestamp records the time a restore was started. The server's time is used for StartTimestamps

.status.progress

Description

Progress holds the total number of bytes of the snapshot and the current number of restored bytes. This can be used to display progress information about the restore

operation.

Type

`object`

Property	Type	Description
`bytesDone`	`integer`	
`totalBytes`	`integer`	

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/podvolumerestores`
  - `DELETE`: delete collection of PodVolumeRestore
  - `GET`: list objects of kind PodVolumeRestore
  - `POST`: create a new PodVolumeRestore
- `/apis/velero.io/v1/namespaces/{namespace}/podvolumerestores/{name}`
  - `DELETE`: delete the specified PodVolumeRestore
  - `GET`: read the specified PodVolumeRestore
  - `PATCH`: partially update the specified PodVolumeRestore
  - `PUT`: replace the specified PodVolumeRestore
- `/apis/velero.io/v1/namespaces/{namespace}/podvolumerestores/{name}/status`
  - `GET`: read status of the specified PodVolumeRestore
  - `PATCH`: partially update status of the specified PodVolumeRestore
  - `PUT`: replace status of the specified PodVolumeRestore

/apis/velero.io/v1/namespaces/{namespace}/podvolumerestores

HTTP method

`DELETE`

Description

delete collection of PodVolumeRestore

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind PodVolumeRestore

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestoreList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new PodVolumeRestore

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`PodVolumeRestore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema
201 - Created	`PodVolumeRestore` schema

HTTP code	Response body
202 - Accepted	`PodVolumeRestore` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/podvolumerestores/{name}

HTTP method

`DELETE`

Description

delete the specified PodVolumeRestore

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified PodVolumeRestore

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified PodVolumeRestore

Query parameters

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

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified PodVolumeRestore

Query parameters

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

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`PodVolumeRestore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema
201 - Created	`PodVolumeRestore` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/podvolumerestores/{name}/status

HTTP method

`GET`

Description

read status of the specified PodVolumeRestore

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified PodVolumeRestore

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified PodVolumeRestore

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`PodVolumeRestore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`PodVolumeRestore` schema
201 - Created	`PodVolumeRestore` schema
401 - Unauthorized	Empty

Restore [velero.io/v1]

Description

Restore is a Velero resource that represents the application of resources from a Velero backup to a target Kubernetes cluster.

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	RestoreSpec defines the specification for a Velero restore.
`status`	`object`	RestoreStatus captures the current status of a Velero restore

.spec

Description

RestoreSpec defines the specification for a Velero restore.

Type

`object`

Property	Type	Description
`backupName`	`string`	BackupName is the unique name of the Velero backup to restore from.
`excludedNamespaces`	`array`	ExcludedNamespaces contains a list of namespaces that are not included in the restore.

Property	Type	Description
`excludedResources`	`array`	ExcludedResources is a slice of resource names that are not included in the restore.
`existingResourcePolicy`	`string`	ExistingResourcePolicy specifies the restore behavior for the Kubernetes resource to be restored
`hooks`	`object`	Hooks represent custom behaviors that should be executed during or post restore.
`includeClusterResources`	`boolean`	IncludeClusterResources specifies whether cluster-scoped resources should be included for consideration in the restore. If null, defaults to true.
`includedNamespaces`	`array`	IncludedNamespaces is a slice of namespace names to include objects from. If empty, all namespaces are included.
`includedResources`	`array`	IncludedResources is a slice of resource names to include in the restore. If empty, all resources in the backup are included.

Property	Type	Description
`itemOperationTimeout`	`string`	ItemOperationTimeout specifies the time used to wait for RestoreItemAction operations The default value is 4 hour.
`labelSelector`	`object`	LabelSelector is a metav1.LabelSelector to filter with when restoring individual objects from the backup. If empty or nil, all objects are included. Optional.
`namespaceMapping`	`object`	NamespaceMapping is a map of source namespace names to target namespace names to restore into. Any source namespaces not included in the map will be restored into namespaces of the same name.
`orLabelSelectors`	`array`	OrLabelSelectors is list of metav1.LabelSelector to filter with when restoring individual objects from the backup. If multiple provided they will be joined by the OR operator. LabelSelector as well as OrLabelSelectors cannot co-exist in restore request, only one of them can be used
`preserveNodePorts`	`boolean`	PreserveNodePorts specifies whether to restore old nodePorts from backup.

Property	Type	Description
`resourceModifier`	`object`	ResourceModifier specifies the reference to JSON resource patches that should be applied to resources before restoration.
`restorePVs`	`boolean`	RestorePVs specifies whether to restore all included PVs from snapshot
`restoreStatus`	`object`	RestoreStatus specifies which resources we should restore the status field. If nil, no objects are included. Optional.
`scheduleName`	`string`	ScheduleName is the unique name of the Velero schedule to restore from. If specified, and BackupName is empty, Velero will restore from the most recent successful backup created from this schedule.
`uploaderConfig`	`object`	UploaderConfig specifies the configuration for the restore.

.spec.excludedNamespaces

Description

ExcludedNamespaces contains a list of namespaces that are not included in the restore.

Type

`array`

.spec.excludedNamespaces[]

Type

`string`

.spec.excludedResources

Description

ExcludedResources is a slice of resource names that are not included in the restore.

Type

`array`

.spec.excludedResources[]

Type

`string`

.spec.hooks

Description

Hooks represent custom behaviors that should be executed during or post restore.

Type

`object`

Property	Type	Description
`resources`	`array`	

.spec.hooks.resources

Type

`array`

.spec.hooks.resources[]

Description

RestoreResourceHookSpec defines one or more RestoreResrouceHooks that should be executed based on the rules defined for namespaces, resources, and label selector.

Type

`object`

Required

`name`

Property	Type	Description
`excludedNamespaces`	`array`	ExcludedNamespaces specifies the namespaces to which this hook spec does not apply.
`excludedResources`	`array`	ExcludedResources specifies the resources to which this hook spec does not apply.
`includedNamespaces`	`array`	IncludedNamespaces specifies the namespaces to which this hook spec applies. If empty, it applies to all namespaces.
`includedResources`	`array`	IncludedResources specifies the resources to which this hook spec applies. If empty, it applies to all resources.
`labelSelector`	`object`	LabelSelector, if specified, filters the resources to which this hook spec applies.

Property	Type	Description
`name`	`string`	Name is the name of this hook.
`postHooks`	`array`	PostHooks is a list of RestoreResourceHooks to execute during and after restoring a resource.

.spec.hooks.resources[].excludedNamespaces

Description

ExcludedNamespaces specifies the namespaces to which this hook spec does not apply.

Type

`array`

.spec.hooks.resources[].excludedNamespaces[]

Type

`string`

.spec.hooks.resources[].excludedResources

Description

ExcludedResources specifies the resources to which this hook spec does not apply.

Type

`array`

.spec.hooks.resources[].excludedResources[]

Type

`string`

.spec.hooks.resources[].includedNamespaces

Description

IncludedNamespaces specifies the namespaces to which this hook spec applies. If empty, it applies to all namespaces.

Type

`array`

.spec.hooks.resources[].includedNamespaces[]

Type

`string`

.spec.hooks.resources[].includedResources

Description

IncludedResources specifies the resources to which this hook spec applies. If empty, it applies to all resources.

Type

`array`

.spec.hooks.resources[].includedResources[]

Type

`string`

.spec.hooks.resources[].labelSelector

Description

LabelSelector, if specified, filters the resources to which this hook spec applies.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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.

.spec.hooks.resources[].labelSelector.matchExpressions

Description

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

Type

`array`

.spec.hooks.resources[].labelSelector.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.hooks.resources[].labelSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.hooks.resources[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.hooks.resources[].labelSelector.matchLabels

Description

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

Type

`object`

.spec.hooks.resources[].postHooks

Description

PostHooks is a list of RestoreResourceHooks to execute during and after restoring a resource.

Type

`array`

.spec.hooks.resources[].postHooks[]

Description

RestoreResourceHook defines a restore hook for a resource.

Type

`object`

Property	Type	Description
`exec`	`object`	Exec defines an exec restore hook.
`init`	`object`	Init defines an init restore hook.

.spec.hooks.resources[].postHooks[].exec

Description

Exec defines an exec restore hook.

Type

`object`

Required

`command`

Property	Type	Description
`command`	`array`	Command is the command and arguments to execute from within a container after a pod has been restored.
`container`	`string`	Container is the container in the pod where the command should be executed. If not specified, the pod's first container is used.
`execTimeout`	`string`	ExecTimeout defines the maximum amount of time Velero should wait for the hook to complete before considering the execution a failure.
`onError`	`string`	OnError specifies how Velero should behave if it encounters an error executing this hook.
`waitForReady`	`boolean`	WaitForReady ensures command will be launched when container is Ready instead of Running.
`waitTimeout`	`string`	WaitTimeout defines the maximum amount of time Velero should wait for the container to be Ready before

Property	Type	Description
		attempting to run the command.

.spec.hooks.resources[].postHooks[].exec.command

Description

Command is the command and arguments to execute from within a container after a pod has been restored.

Type

`array`

.spec.hooks.resources[].postHooks[].exec.command[]

Type

`string`

.spec.hooks.resources[].postHooks[].init

Description

Init defines an init restore hook.

Type

`object`

Property	Type	Description
`initContainers`	`array`	InitContainers is list of init containers to be added to a pod during its restore.

Property	Type	Description
`timeout`	`string`	Timeout defines the maximum amount of time Velero should wait for the initContainers to complete.

.spec.hooks.resources[].postHooks[].init.initContainers

Description

InitContainers is list of init containers to be added to a pod during its restore.

Type

`array`

.spec.hooks.resources[].postHooks[].init.initContainers[]

Type

`object`

.spec.includedNamespaces

Description

IncludedNamespaces is a slice of namespace names to include objects from. If empty, all namespaces are included.

Type

`array`

.spec.includedNamespaces[]

Type

`string`

.spec.includedResources

Description

IncludedResources is a slice of resource names to include in the restore. If empty, all resources in the backup are included.

Type

`array`

.spec.includedResources[]

Type

`string`

.spec.labelSelector

Description

LabelSelector is a metav1.LabelSelector to filter with when restoring individual objects from the backup. If empty or nil, all objects are included. Optional.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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.

.spec.labelSelector.matchExpressions

Description

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

Type

`array`

.spec.labelSelector.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.labelSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.labelSelector.matchLabels

Description

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

Type

`object`

.spec.namespaceMapping

Description

NamespaceMapping is a map of source namespace names to target namespace names to restore into. Any source namespaces not included in the map will be restored into namespaces of the same name.

Type

`object`

.spec.orLabelSelectors

Description

OrLabelSelectors is list of metav1.LabelSelector to filter with when restoring individual objects from the backup. If multiple provided they will be joined by the OR operator. LabelSelector as well as OrLabelSelectors cannot co-exist in restore request, only one of them can be used

Type

`array`

.spec.orLabelSelectors[]

Description

A label selector is a label query over a set of resources. The result of matchLabels and matchExpressions are ANDed. An empty label selector matches all objects. A null label selector matches no objects.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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.

.spec.orLabelSelectors[].matchExpressions

Description

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

Type

`array`

.spec.orLabelSelectors[].matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.orLabelSelectors[].matchExpressions[].values

Description

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

Type

`array`

.spec.orLabelSelectors[].matchExpressions[].values[]

Type

`string`

.spec.orLabelSelectors[].matchLabels

Description

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

Type

`object`

.spec.resourceModifier

Description

ResourceModifier specifies the reference to JSON resource patches that should be applied to resources before restoration.

Type

`object`

Required

`kind` `name`

Property	Type	Description
`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`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.restoreStatus

Description

RestoreStatus specifies which resources we should restore the status field. If nil, no objects are included. Optional.

Type

`object`

Property	Type	Description
`excludedResources`	`array`	ExcludedResources specifies the resources to which will not restore the status.
`includedResources`	`array`	IncludedResources specifies the resources to which will restore the status. If empty, it applies to all resources.

.spec.restoreStatus.excludedResources

Description

ExcludedResources specifies the resources to which will not restore the status.

Type

`array`

.spec.restoreStatus.excludedResources[]

Type

`string`

.spec.restoreStatus.includedResources

Description

IncludedResources specifies the resources to which will restore the status. If empty, it applies to all resources.

Type

`array`

.spec.restoreStatus.includedResources[]

Type

`string`

.spec.uploaderConfig

Description

UploaderConfig specifies the configuration for the restore.

Type

`object`

Property	Type	Description
`parallelFilesDownload`	`integer`	ParallelFilesDownload is the concurrency number setting for restore.
`writeSparseFiles`	`boolean`	WriteSparseFiles is a flag to indicate whether write files sparsely or not.

.status

Description

RestoreStatus captures the current status of a Velero restore

Type

`object`

Property	Type	Description
`completionTimestamp`	`string`	CompletionTimestamp records the time the restore operation was completed. Completion time is recorded even on failed restore. The server's time is used for StartTimestamps
`errors`	`integer`	Errors is a count of all error messages that were generated during execution of the restore. The actual errors are stored in object storage.

Property	Type	Description
`failureReason`	`string`	FailureReason is an error that caused the entire restore to fail.
`hookStatus`	`object`	HookStatus contains information about the status of the hooks.
`phase`	`string`	Phase is the current state of the Restore
`progress`	`object`	Progress contains information about the restore's execution progress. Note that this information is best-effort only -- if Velero fails to update it during a restore for any reason, it may be inaccurate/stale.
`restoreItemOperationsAttempted`	`integer`	RestoreItemOperationsAttempted is the total number of attempted async RestoreItemAction operations for this restore.
`restoreItemOperationsCompleted`	`integer`	RestoreItemOperationsCompleted is the total number of successfully completed async RestoreItemAction operations for this restore.

Property	Type	Description
`restoreItemOperationsFailed`	`integer`	RestoreItemOperationsFailed is the total number of async RestoreItemAction operations for this restore which ended with an error.
`startTimestamp`	`string`	StartTimestamp records the time the restore operation was started. The server's time is used for StartTimestamps
`validationErrors`	`array`	ValidationErrors is a slice of all validation errors (if applicable)
`warnings`	`integer`	Warnings is a count of all warning messages that were generated during execution of the restore. The actual warnings are stored in object storage.

.status.hookStatus

Description

HookStatus contains information about the status of the hooks.

Type

`object`

Property	Type	Description
`hooksAttempted`	`integer`	HooksAttempted is the total number of attempted hooks Specifically, HooksAttempted represents the number of hooks that failed to execute and the number of hooks that executed successfully.
`hooksFailed`	`integer`	HooksFailed is the total number of hooks which ended with an error

.status.progress

Description

Progress contains information about the restore's execution progress. Note that this information is best-effort only -- if Velero fails to update it during a restore for any reason, it may be inaccurate/stale.

Type

`object`

Property	Type	Description
`itemsRestored`	`integer`	ItemsRestored is the number of items that have actually been restored so far
`totalItems`	`integer`	TotalItems is the total number of items to be restored. This number may change throughout the execution of the restore due to plugins that return additional related items to restore

.status.validationErrors

Description

ValidationErrors is a slice of all validation errors (if applicable)

Type

`array`

.status.validationErrors[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/restores`
  - `DELETE`: delete collection of Restore
  - `GET`: list objects of kind Restore
  - `POST`: create a new Restore
- `/apis/velero.io/v1/namespaces/{namespace}/restores/{name}`
  - `DELETE`: delete the specified Restore
  - `GET`: read the specified Restore
  - `PATCH`: partially update the specified Restore
  - `PUT`: replace the specified Restore
- `/apis/velero.io/v1/namespaces/{namespace}/restores/{name}/status`
  - `GET`: read status of the specified Restore
  - `PATCH`: partially update status of the specified Restore
  - `PUT`: replace status of the specified Restore

/apis/velero.io/v1/namespaces/{namespace}/restores

HTTP method

`DELETE`

Description

delete collection of Restore

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind Restore

HTTP responses

HTTP code	Response body
200 - OK	`RestoreList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new Restore

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`Restore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema
201 - Created	`Restore` schema

HTTP code	Response body
202 - Accepted	`Restore` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/restores/{name}

HTTP method

`DELETE`

Description

delete the specified Restore

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified Restore

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified Restore

Query parameters

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

Parameter	Type	Description
		will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified Restore

Query parameters

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

Parameter	Type	Description
		request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`Restore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema
201 - Created	`Restore` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/restores/{name}/status

HTTP method

`GET`

Description

read status of the specified Restore

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified Restore

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified Restore

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`Restore` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Restore` schema
201 - Created	`Restore` schema
401 - Unauthorized	Empty

Schedule [velero.io/v1]

Description

Schedule is a Velero resource that represents a pre-scheduled or periodic Backup that should be run.

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	ScheduleSpec defines the specification for a Velero schedule
`status`	`object`	ScheduleStatus captures the current state of a Velero schedule

.spec

Description

ScheduleSpec defines the specification for a Velero schedule

Type

`object`

Required

`schedule` `template`

Property	Type	Description
`paused`	`boolean`	Paused specifies whether the schedule is paused or not

Property	Type	Description
`schedule`	`string`	Schedule is a Cron expression defining when to run the Backup.
`skipImmediately`	`boolean`	SkipImmediately specifies whether to skip backup if schedule is due immediately from `schedule.status.lastBackup` timestamp when schedule is unpaused or if schedule is new. If true, backup will be skipped immediately when schedule is unpaused if it is due based on .Status.LastBackupTimestamp or schedule is new, and will run at next schedule time. If false, backup will not be skipped immediately when schedule is unpaused, but will run at next schedule time. If empty, will follow server configuration (default: false).
`template`	`object`	Template is the definition of the Backup to be run on the provided schedule
`useOwnerReferencesInBackup`	`boolean`	UseOwnerReferencesBackup specifies whether to use OwnerReferences on backups created by this Schedule.

.spec.template

Description

Template is the definition of the Backup to be run on the provided schedule

Type

`object`

Property	Type	Description
`csiSnapshotTimeout`	`string`	CSISnapshotTimeout specifies the time used to wait for CSI VolumeSnapshot status turns to ReadyToUse during creation, before returning error as timeout. The default value is 10 minute
`datamover`	`string`	DataMover specifies the data mover to be used by the backup. If DataMover is "" or "velero", the built-in data mover will be used.
`defaultVolumesToFsBackup`	`boolean`	DefaultVolumesToFsBackup specifies whether pod volume file system backup should be used for all volumes by default.
`defaultVolumesToRestic`	`boolean`	DefaultVolumesToRestic specifies whether restic should be used to take a backup of all pod volumes by default. Deprecated: this field is no longer used and will be removed entirely in future. Use DefaultVolumesToFsBackup instead.

Property	Type	Description
`excludedClusterScopedResources`	`array`	ExcludedClusterScopedResources is a slice of cluster-scoped resource type names to exclude from the backup. If set to "*", all cluster-scoped resource types are excluded. The default value is empty.
`excludedNamespaceScopedResources`	`array`	ExcludedNamespaceScopedResources is a slice of namespace-scoped resource type names to exclude from the backup. If set to "*", all namespace-scoped resource types are excluded. The default value is empty.
`excludedNamespaces`	`array`	ExcludedNamespaces contains a list of namespaces that are not included in the backup.
`excludedResources`	`array`	ExcludedResources is a slice of resource names that are not included in the backup.
`hooks`	`object`	Hooks represent custom behaviors that should be executed at different phases of the backup.

Property	Type	Description
`includeClusterResources`	`boolean`	IncludeClusterResources specifies whether cluster-scoped resources should be included for consideration in the backup.
`includedClusterScopedResources`	`array`	IncludedClusterScopedResources is a slice of cluster-scoped resource type names to include in the backup. If set t "*", all cluster-scoped resource types are included. The default value is empty, which means only related cluster-scoped resources are included.
`includedNamespaceScopedResources`	`array`	IncludedNamespaceScopedResources is a slice of namespace-scoped resource type names to include in the backup. The default value is "*".
`includedNamespaces`	`array`	IncludedNamespaces is a slice of namespace names to include objects from. If empty, all namespaces are included.
`includedResources`	`array`	IncludedResources is a slice of resource names to include in the backup. If empty, all resources are included.

Property	Type	Description
`itemOperationTimeout`	`string`	ItemOperationTimeout specifies the time used to wait for asynchronous BackupItemAction operations The default value is 4 hour.
`labelSelector`	`object`	LabelSelector is a metav1.LabelSelector to filter with when adding individual objects to the backup. If empty or nil, all objects are included. Optional.
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`orLabelSelectors`	`array`	OrLabelSelectors is list of metav1.LabelSelector to filter with when adding individual objects to the backup. If multiple provided they will be joined by the OR operator. LabelSelector as well as OrLabelSelectors cannot co-exist in backup request, only one of them can be used.
`orderedResources`	`object`	OrderedResources specifies the backup order of resources of specific Kind. The map key is the resource name and value is a list of object

Property	Type	Description
		names separated by commas. Each resource name has format "namespace/objectname". For cluster resources, simply use "objectname".
`resourcePolicy`	`object`	ResourcePolicy specifies the referenced resource policies that backup should follow
`snapshotMoveData`	`boolean`	SnapshotMoveData specifies whether snapshot data should be moved
`snapshotVolumes`	`boolean`	SnapshotVolumes specifies whether to take snapshots of any PV's referenced in the set of objects included in the Backup.
`storageLocation`	`string`	StorageLocation is a string containing the name of a BackupStorageLocation where the backup should be stored.
`ttl`	`string`	TTL is a time.Duration-parseable string describing how long the Backup should be retained for.

Property	Type	Description
`uploaderConfig`	`object`	UploaderConfig specifies the configuration for the uploader.
`volumeSnapshotLocations`	`array`	VolumeSnapshotLocations is a list containing names of VolumeSnapshotLocations associated with this backup.

.spec.template.excludedClusterScopedResources

Description

ExcludedClusterScopedResources is a slice of cluster-scoped resource type names to exclude from the backup. If set to "*", all cluster-scoped resource types are excluded. The default value is empty.

Type

`array`

.spec.template.excludedClusterScopedResources[]

Type

`string`

.spec.template.excludedNamespaceScopedResources

Description

ExcludedNamespaceScopedResources is a slice of namespace-scoped resource type names to exclude from the backup. If set to "*", all namespace-scoped resource types are excluded. The default value is empty.

Type

`array`

.spec.template.excludedNamespaceScopedResources[]

Type

`string`

.spec.template.excludedNamespaces

Description

ExcludedNamespaces contains a list of namespaces that are not included in the backup.

Type

`array`

.spec.template.excludedNamespaces[]

Type

`string`

.spec.template.excludedResources

Description

ExcludedResources is a slice of resource names that are not included in the backup.

Type

`array`

.spec.template.excludedResources[]

Type

`string`

.spec.template.hooks

Description

Hooks represent custom behaviors that should be executed at different phases of the backup.

Type

`object`

Property	Type	Description
`resources`	`array`	Resources are hooks that should be executed when backing up individual instances of a resource.

.spec.template.hooks.resources

Description

Resources are hooks that should be executed when backing up individual instances of a resource.

Type

`array`

.spec.template.hooks.resources[]

Description

BackupResourceHookSpec defines one or more BackupResourceHooks that should be executed based on the rules defined for namespaces, resources, and label selector.

Type

`object`

Required

`name`

Property	Type	Description
`excludedNamespaces`	`array`	ExcludedNamespaces specifies the namespaces to which this hook spec does not apply.
`excludedResources`	`array`	ExcludedResources specifies the resources to which this hook spec does not apply.
`includedNamespaces`	`array`	IncludedNamespaces specifies the namespaces to which this hook spec applies. If empty, it applies to all namespaces.
`includedResources`	`array`	IncludedResources specifies the resources to which this hook spec applies. If empty, it applies to all resources.
`labelSelector`	`object`	LabelSelector, if specified, filters the resources to which this hook spec applies.
`name`	`string`	Name is the name of this hook.
`post`	`array`	PostHooks is a list of BackupResourceHooks to execute after storing the item in the backup. These are executed after all "additional items" from item actions are processed.

Property	Type	Description
`pre`	`array`	PreHooks is a list of BackupResourceHooks to execute prior to storing the item in the backup. These are executed before any "additional items" from item actions are processed.

.spec.template.hooks.resources[].excludedNamespaces

Description

ExcludedNamespaces specifies the namespaces to which this hook spec does not apply.

Type

`array`

.spec.template.hooks.resources[].excludedNamespaces[]

Type

`string`

.spec.template.hooks.resources[].excludedResources

Description

ExcludedResources specifies the resources to which this hook spec does not apply.

Type

`array`

.spec.template.hooks.resources[].excludedResources[]

Type

`string`

.spec.template.hooks.resources[].includedNamespaces

Description

IncludedNamespaces specifies the namespaces to which this hook spec applies. If empty, it applies to all namespaces.

Type

`array`

.spec.template.hooks.resources[].includedNamespaces[]

Type

`string`

.spec.template.hooks.resources[].includedResources

Description

IncludedResources specifies the resources to which this hook spec applies. If empty, it applies to all resources.

Type

`array`

.spec.template.hooks.resources[].includedResources[]

Type

`string`

.spec.template.hooks.resources[].labelSelector

Description

LabelSelector, if specified, filters the resources to which this hook spec applies.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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.

.spec.template.hooks.resources[].labelSelector.matchExpressions

Description

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

Type

`array`

.spec.template.hooks.resources[].labelSelector.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.template.hooks.resources[].labelSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.template.hooks.resources[].labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.hooks.resources[].labelSelector.matchLabels

Description

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

Type

`object`

.spec.template.hooks.resources[].post

Description

PostHooks is a list of BackupResourceHooks to execute after storing the item in the backup. These are executed after all "additional items" from item actions are processed.

Type

`array`

.spec.template.hooks.resources[].post[]

Description

BackupResourceHook defines a hook for a resource.

Type

`object`

Required

`exec`

Property	Type	Description
`exec`	`object`	Exec defines an exec hook.

.spec.template.hooks.resources[].post[].exec

Description

Exec defines an exec hook.

Type

`object`

Required

`command`

Property	Type	Description
`command`	`array`	Command is the command and arguments to execute.
`container`	`string`	Container is the container in the pod where the command should be executed. If not specified, the pod's first container is used.
`onError`	`string`	OnError specifies how Velero should behave if it encounters an error executing this hook.
`timeout`	`string`	Timeout defines the maximum amount of time Velero should wait for the hook to complete before considering the execution a failure.

.spec.template.hooks.resources[].post[].exec.command

Description

Command is the command and arguments to execute.

Type

`array`

.spec.template.hooks.resources[].post[].exec.command[]

Type

`string`

.spec.template.hooks.resources[].pre

Description

PreHooks is a list of BackupResourceHooks to execute prior to storing the item in the backup. These are executed before any "additional items" from item actions are processed.

Type

`array`

.spec.template.hooks.resources[].pre[]

Description

BackupResourceHook defines a hook for a resource.

Type

`object`

Required

`exec`

Property	Type	Description
`exec`	`object`	Exec defines an exec hook.

.spec.template.hooks.resources[].pre[].exec

Description

Exec defines an exec hook.

Type

`object`

Required

`command`

Property	Type	Description
`command`	`array`	Command is the command and arguments to execute.
`container`	`string`	Container is the container in the pod where the command should be executed. If not specified, the pod's first container is used.
`onError`	`string`	OnError specifies how Velero should behave if it encounters an error executing this hook.
`timeout`	`string`	Timeout defines the maximum amount of time Velero should wait for the hook to complete before considering the execution a failure.

.spec.template.hooks.resources[].pre[].exec.command

Description

Command is the command and arguments to execute.

Type

`array`

.spec.template.hooks.resources[].pre[].exec.command[]

Type

`string`

.spec.template.includedClusterScopedResources

Description

IncludedClusterScopedResources is a slice of cluster-scoped resource type names to include in the backup. If set to "*", all cluster-scoped resource types are included. The default value is empty, which means only related cluster-scoped resources are included.

Type

`array`

.spec.template.includedClusterScopedResources[]

Type

`string`

.spec.template.includedNamespaceScopedResources

Description

IncludedNamespaceScopedResources is a slice of namespace-scoped resource type names to include in the backup. The default value is "*".

Type

`array`

.spec.template.includedNamespaceScopedResources[]

Type

`string`

.spec.template.includedNamespaces

Description

IncludedNamespaces is a slice of namespace names to include objects from. If empty, all namespaces are included.

Type

`array`

.spec.template.includedNamespaces[]

Type

`string`

.spec.template.includedResources

Description

IncludedResources is a slice of resource names to include in the backup. If empty, all resources are included.

Type

`array`

.spec.template.includedResources[]

Type

`string`

.spec.template.labelSelector

Description

LabelSelector is a metav1.LabelSelector to filter with when adding individual objects to the backup. If empty or nil, all objects are included. Optional.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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.

.spec.template.labelSelector.matchExpressions

Description

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

Type

`array`

.spec.template.labelSelector.matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

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

.spec.template.labelSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.template.labelSelector.matchExpressions[].values[]

Type

`string`

.spec.template.labelSelector.matchLabels

Description

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

Type

`object`

.spec.template.orLabelSelectors

Description

OrLabelSelectors is list of metav1.LabelSelector to filter with when adding individual objects to the backup. If multiple provided they will be joined by the OR operator. LabelSelector as well as OrLabelSelectors cannot co-exist in backup request, only one of them can be used.

Type

`array`

.spec.template.orLabelSelectors[]

Description

A label selector is a label query over a set of resources. The result of matchLabels and matchExpressions are ANDed. An empty label selector matches all objects. A null label selector matches no objects.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	matchExpressions is a list of label selector requirements. The requirements are ANDed.
`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

Property	Type	Description
		"key", the operator is "In", and the values array contains only "value". The requirements are ANDed.

.spec.template.orLabelSelectors[].matchExpressions

Description

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

Type

`array`

.spec.template.orLabelSelectors[].matchExpressions[]

Description

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

Type

`object`

Required

`key` `operator`

Property	Type	Description
`key`	`string`	key is the label key that the selector applies to.
`operator`	`string`	operator represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists and DoesNotExist.

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

.spec.template.orLabelSelectors[].matchExpressions[].values

Description

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

Type

`array`

.spec.template.orLabelSelectors[].matchExpressions[].values[]

Type

`string`

.spec.template.orLabelSelectors[].matchLabels

Description

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

Type

`object`

.spec.template.orderedResources

Description

OrderedResources specifies the backup order of resources of specific Kind. The map key is the resource name and value is a list of object names separated by commas. Each resource name has format "namespace/objectname". For cluster resources, simply use "objectname".

Type

`object`

.spec.template.resourcePolicy

Description

ResourcePolicy specifies the referenced resource policies that backup should follow

Type

`object`

Required

`kind` `name`

Property	Type	Description
`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`	Kind is the type of resource being referenced
`name`	`string`	Name is the name of resource being referenced

.spec.template.uploaderConfig

Description

UploaderConfig specifies the configuration for the uploader.

Type

`object`

Property	Type	Description
`parallelFilesUpload`	`integer`	ParallelFilesUpload is the number of files parallel uploads to perform when using the uploader.

.spec.template.volumeSnapshotLocations

Description

VolumeSnapshotLocations is a list containing names of VolumeSnapshotLocations associated with this backup.

Type

`array`

.spec.template.volumeSnapshotLocations[]

Type

`string`

.status

Description

ScheduleStatus captures the current state of a Velero schedule

Type

`object`

Property	Type	Description
`lastBackup`	`string`	LastBackup is the last time a Backup was run for this Schedule schedule
`lastSkipped`	`string`	LastSkipped is the last time a Schedule was skipped
`phase`	`string`	Phase is the current phase of the Schedule
`validationErrors`	`array`	ValidationErrors is a slice of all validation errors (if applicable)

.status.validationErrors

Description

ValidationErrors is a slice of all validation errors (if applicable)

Type

`array`

.status.validationErrors[]

Type

`string`

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/schedules`
  - `DELETE`: delete collection of Schedule
  - `GET`: list objects of kind Schedule
  - `POST`: create a new Schedule
- `/apis/velero.io/v1/namespaces/{namespace}/schedules/{name}`
  - `DELETE`: delete the specified Schedule
  - `GET`: read the specified Schedule
  - `PATCH`: partially update the specified Schedule
  - `PUT`: replace the specified Schedule
- `/apis/velero.io/v1/namespaces/{namespace}/schedules/{name}/status`
  - `GET`: read status of the specified Schedule
  - `PATCH`: partially update status of the specified Schedule
  - `PUT`: replace status of the specified Schedule

/apis/velero.io/v1/namespaces/{namespace}/schedules

HTTP method

`DELETE`

Description

delete collection of Schedule

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind Schedule

HTTP responses

HTTP code	Response body
200 - OK	`ScheduleList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new Schedule

Query parameters

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

Parameter	Type	Description
		fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`Schedule` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema
201 - Created	`Schedule` schema
202 - Accepted	`Schedule` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/schedules/{name}

HTTP method

`DELETE`

Description

delete the specified Schedule

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will

Parameter	Type	Description
		result in an error response and no further processing of the request. Valid values are: - All: all dry run stages will be processed

HTTP responses

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

HTTP method

`GET`

Description

read the specified Schedule

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified Schedule

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified Schedule

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`Schedule` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema

HTTP code	Response body
201 - Created	`Schedule` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/schedules/{name}/status

HTTP method

`GET`

Description

read status of the specified Schedule

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified Schedule

Query parameters

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

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

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified Schedule

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

Body parameters

Parameter	Type	Description
`body`	`Schedule` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`Schedule` schema
201 - Created	`Schedule` schema
401 - Unauthorized	Empty

ServerStatusRequest [velero.io/v1]

Description

ServerStatusRequest is a request to access current status information about the Velero server.

Type

`object`

Specification

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

Property	Type	Description
		https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	ServerStatusRequestSpec is the specification for a ServerStatusRequest.
`status`	`object`	ServerStatusRequestStatus is the current status of a ServerStatusRequest.

.spec

Description

ServerStatusRequestSpec is the specification for a ServerStatusRequest.

Type

`object`

.status

Description

ServerStatusRequestStatus is the current status of a ServerStatusRequest.

Type

`object`

Property	Type	Description
`phase`	`string`	Phase is the current lifecycle phase of the ServerStatusRequest.
`plugins`	`array`	Plugins list information about the plugins running on the Velero server
`processedTimestamp`	`string`	ProcessedTimestamp is when the ServerStatusRequest was processed by the ServerStatusRequestController.
`serverVersion`	`string`	ServerVersion is the Velero server version.

.status.plugins

Description

Plugins list information about the plugins running on the Velero server

Type

`array`

.status.plugins[]

Description

PluginInfo contains attributes of a Velero plugin

Type

`object`

Required

`kind` `name`

Property	Type	Description
`kind`	`string`	
`name`	`string`	

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/serverstatusrequests`
  - `DELETE`: delete collection of ServerStatusRequest
  - `GET`: list objects of kind ServerStatusRequest
  - `POST`: create a new ServerStatusRequest
- `/apis/velero.io/v1/namespaces/{namespace}/serverstatusrequests/{name}`
  - `DELETE`: delete the specified ServerStatusRequest
  - `GET`: read the specified ServerStatusRequest
  - `PATCH`: partially update the specified ServerStatusRequest
  - `PUT`: replace the specified ServerStatusRequest
- `/apis/velero.io/v1/namespaces/{namespace}/serverstatusrequests/{name}/status`
  - `GET`: read status of the specified ServerStatusRequest
  - `PATCH`: partially update status of the specified ServerStatusRequest
  - `PUT`: replace status of the specified ServerStatusRequest

/apis/velero.io/v1/namespaces/{namespace}/serverstatusrequests

HTTP method

`DELETE`

Description

delete collection of ServerStatusRequest

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind ServerStatusRequest

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequestList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new ServerStatusRequest

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun directive will result in an error response and no further

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

Body parameters

Parameter	Type	Description
`body`	`ServerStatusRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
201 - Created	`ServerStatusRequest` schema
202 - Accepted	`ServerStatusRequest` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/serverstatusrequests/{name}

HTTP method

`DELETE`

Description

delete the specified ServerStatusRequest

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified ServerStatusRequest

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified ServerStatusRequest

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified ServerStatusRequest

Query parameters

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

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

Body parameters

Parameter	Type	Description
`body`	`ServerStatusRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
201 - Created	`ServerStatusRequest` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/serverstatusrequests/{name}/status

HTTP method

`GET`

Description

read status of the specified ServerStatusRequest

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified ServerStatusRequest

Query parameters

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

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified ServerStatusRequest

Query parameters

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

Parameter	Type	Description
		and for each duplicate field that is encountered. The request will still succeed if there are no other errors, and will only persist the last of any duplicate fields. This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`ServerStatusRequest` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`ServerStatusRequest` schema
201 - Created	`ServerStatusRequest` schema
401 - Unauthorized	Empty

VolumeSnapshotLocation [velero.io/v1]

Description

VolumeSnapshotLocation is a location where Velero stores volume snapshots.

Type

`object`

Specification

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

Property	Type	Description
`kind`	`string`	Kind is a string value representing the REST resource this object represents. Servers may infer this from the endpoint the client submits requests to. Cannot be updated. In CamelCase. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds ↗
`metadata`	`ObjectMeta`	ObjectMeta is metadata that all persisted resources must have, which includes all objects users must create.
`spec`	`object`	VolumeSnapshotLocationSpec defines the specification for a Velero VolumeSnapshotLocation.
`status`	`object`	VolumeSnapshotLocationStatus describes the current status of a Velero VolumeSnapshotLocation.

.spec

Description

VolumeSnapshotLocationSpec defines the specification for a Velero VolumeSnapshotLocation.

Type

`object`

Required

`provider`

Property	Type	Description
`config`	`object`	Config is for provider-specific configuration fields.
`credential`	`object`	Credential contains the credential information intended to be used with this location
`provider`	`string`	Provider is the provider of the volume storage.

.spec.config

Description

Config is for provider-specific configuration fields.

Type

`object`

.spec.credential

Description

Credential contains the credential information intended to be used with this location

Type

`object`

Required

`key`

Property	Type	Description
`key`	`string`	The key of the secret to select from. Must be a valid secret key.
`name`	`string`	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names ↗ TODO: Add other useful fields. apiVersion, kind, uid?
`optional`	`boolean`	Specify whether the Secret or its key must be defined

.status

Description

VolumeSnapshotLocationStatus describes the current status of a Velero VolumeSnapshotLocation.

Type

`object`

Property	Type	Description
`phase`	`string`	VolumeSnapshotLocationPhase is the lifecycle phase of a Velero VolumeSnapshotLocation.

API Endpoints

The following API endpoints are available:

- `/apis/velero.io/v1/namespaces/{namespace}/volumesnapshotlocations`
  - `DELETE`: delete collection of VolumeSnapshotLocation
  - `GET`: list objects of kind VolumeSnapshotLocation
  - `POST`: create a new VolumeSnapshotLocation
- `/apis/velero.io/v1/namespaces/{namespace}/volumesnapshotlocations/{name}`
  - `DELETE`: delete the specified VolumeSnapshotLocation
  - `GET`: read the specified VolumeSnapshotLocation
  - `PATCH`: partially update the specified VolumeSnapshotLocation
  - `PUT`: replace the specified VolumeSnapshotLocation
- `/apis/velero.io/v1/namespaces/{namespace}/volumesnapshotlocations/{name}/status`
  - `GET`: read status of the specified VolumeSnapshotLocation
  - `PATCH`: partially update status of the specified VolumeSnapshotLocation
  - `PUT`: replace status of the specified VolumeSnapshotLocation

/apis/velero.io/v1/namespaces/{namespace}/volumesnapshotlocations

HTTP method

`DELETE`

Description

delete collection of VolumeSnapshotLocation

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind VolumeSnapshotLocation

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocationList` schema
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new VolumeSnapshotLocation

Query parameters

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

Parameter	Type	Description
		This is the default in v1.23+ - Strict: This will fail the request with a BadRequest error if any unknown fields would be dropped from the object, or if any duplicate fields are present. The error returned from the server will contain all unknown and duplicate fields encountered.

Body parameters

Parameter	Type	Description
`body`	`VolumeSnapshotLocation` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema
201 - Created	`VolumeSnapshotLocation` schema
202 - Accepted	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/volumesnapshotlocations/{name}

HTTP method

`DELETE`

Description

delete the specified VolumeSnapshotLocation

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified VolumeSnapshotLocation

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update the specified VolumeSnapshotLocation

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace the specified VolumeSnapshotLocation

Query parameters

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

Body parameters

Parameter	Type	Description
`body`	`VolumeSnapshotLocation` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema

HTTP code	Response body
201 - Created	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty

/apis/velero.io/v1/namespaces/{namespace}/volumesnapshotlocations/{name}/status

HTTP method

`GET`

Description

read status of the specified VolumeSnapshotLocation

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty

HTTP method

`PATCH`

Description

partially update status of the specified VolumeSnapshotLocation

Query parameters

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

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

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty

HTTP method

`PUT`

Description

replace status of the specified VolumeSnapshotLocation

Query parameters

Parameter	Type	Description
`dryRun`	`string`	When present, indicates that modifications should not be persisted. An invalid or unrecognized dryRun

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

Body parameters

Parameter	Type	Description
`body`	`VolumeSnapshotLocation` schema	`application/json` formatted

HTTP responses

HTTP code	Response body
200 - OK	`VolumeSnapshotLocation` schema
201 - Created	`VolumeSnapshotLocation` schema
401 - Unauthorized	Empty