OpenShift Container Platform 4.18

Storage APIs

Reference guide for storage APIs

Last Updated: 2025-07-25

OpenShift Container Platform 4.18 Storage APIs

Reference guide for storage APIs

Legal Notice

Copyright © 2025 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document describes the OpenShift Container Platform storage API objects and their detailed specifications.

Table of Contents

CHAPTER 1. STORAGE APIS 6
1.1. CSIDRIVER [STORAGE.K8S.IO/V1] 6
1.2. CSINODE [STORAGE.K8S.IO/V1] 6
1.3. CSISTORAGECAPACITY [STORAGE.K8S.IO/V1] 6
1.4. PERSISTENTVOLUME [V1] 6
1.5. PERSISTENTVOLUMECLAIM [V1] 7
1.6. STORAGECLASS [STORAGE.K8S.IO/V1] 7
1.7. STORAGESTATE [MIGRATION.K8S.IO/V1ALPHA1] 7
1.8. STORAGEVERSIONMIGRATION [MIGRATION.K8S.IO/V1ALPHA1] 7
1.9. VOLUMEATTACHMENT [STORAGE.K8S.IO/V1] 7
1.10. VOLUMESNAPSHOT [SNAPSHOT.STORAGE.K8S.IO/V1] 8
1.11. VOLUMESNAPSHOTCLASS [SNAPSHOT.STORAGE.K8S.IO/V1] 8
1.12. VOLUMESNAPSHOTCONTENT [SNAPSHOT.STORAGE.K8S.IO/V1] 8

CHAPTER 2. CSIDRIVER [STORAGE.K8S.IO/V1] 9
2.1. SPECIFICATION 9
2.1.1. .spec 10
2.1.2. .spec.tokenRequests 15
2.1.3. .spec.tokenRequests[] 16
2.2. API ENDPOINTS 16
2.2.1. /apis/storage.k8s.io/v1/csidrivers 17
2.2.2. /apis/storage.k8s.io/v1/watch/csidrivers 18
2.2.3. /apis/storage.k8s.io/v1/csidrivers/{name} 19
2.2.4. /apis/storage.k8s.io/v1/watch/csidrivers/{name} 22

CHAPTER 3. CSINODE [STORAGE.K8S.IO/V1] 23
3.1. SPECIFICATION 23
3.1.1. .spec 24
3.1.2. .spec.drivers 24
3.1.3. .spec.drivers[] 24
3.1.4. .spec.drivers[].allocatable 26
3.2. API ENDPOINTS 26
3.2.1. /apis/storage.k8s.io/v1/csinodes 27
3.2.2. /apis/storage.k8s.io/v1/watch/csinodes 28
3.2.3. /apis/storage.k8s.io/v1/csinodes/{name} 29
3.2.4. /apis/storage.k8s.io/v1/watch/csinodes/{name} 32

CHAPTER 4. CSISTORAGECAPACITY [STORAGE.K8S.IO/V1] 33
4.1. SPECIFICATION 33
4.2. API ENDPOINTS 35
4.2.1. /apis/storage.k8s.io/v1/csistoragecapacities 36
4.2.2. /apis/storage.k8s.io/v1/watch/csistoragecapacities 36
4.2.3. /apis/storage.k8s.io/v1/namespaces/{namespace}/csistoragecapacities 37
4.2.4. /apis/storage.k8s.io/v1/watch/namespaces/{namespace}/csistoragecapacities 39
4.2.5. /apis/storage.k8s.io/v1/namespaces/{namespace}/csistoragecapacities/{name} 39
4.2.6. /apis/storage.k8s.io/v1/watch/namespaces/{namespace}/csistoragecapacities/{name} 42

CHAPTER 5. PERSISTENTVOLUME [V1] 44
5.1. SPECIFICATION 44
5.1.1. .spec 45
5.1.2. .spec.awsElasticBlockStore 50
5.1.3. .spec.azureDisk 51

5.1.4. .spec.azureFile 52
5.1.5. .spec.cephfs 53
5.1.6. .spec.cephfs.secretRef 54
5.1.7. .spec.cinder 55
5.1.8. .spec.cinder.secretRef 55
5.1.9. .spec.claimRef 56
5.1.10. .spec.csi 57
5.1.11. .spec.csi.controllerExpandSecretRef 58
5.1.12. .spec.csi.controllerPublishSecretRef 59
5.1.13. .spec.csi.nodeExpandSecretRef 59
5.1.14. .spec.csi.nodePublishSecretRef 60
5.1.15. .spec.csi.nodeStageSecretRef 60
5.1.16. .spec.fc 60
5.1.17. .spec.flexVolume 61
5.1.18. .spec.flexVolume.secretRef 62
5.1.19. .spec.flocker 62
5.1.20. .spec.gcePersistentDisk 63
5.1.21. .spec.glusterfs 64
5.1.22. .spec.hostPath 65
5.1.23. .spec.iscsi 66
5.1.24. .spec.iscsi.secretRef 68
5.1.25. .spec.local 68
5.1.26. .spec.nfs 69
5.1.27. .spec.nodeAffinity 70
5.1.28. .spec.nodeAffinity.required 70
5.1.29. .spec.nodeAffinity.required.nodeSelectorTerms 70
5.1.30. .spec.nodeAffinity.required.nodeSelectorTerms[] 70
5.1.31. .spec.nodeAffinity.required.nodeSelectorTerms[].matchExpressions 71
5.1.32. .spec.nodeAffinity.required.nodeSelectorTerms[].matchExpressions[] 71
5.1.33. .spec.nodeAffinity.required.nodeSelectorTerms[].matchFields 72
5.1.34. .spec.nodeAffinity.required.nodeSelectorTerms[].matchFields[] 72
5.1.35. .spec.photonPersistentDisk 73
5.1.36. .spec.portworxVolume 74
5.1.37. .spec.quobyte 74
5.1.38. .spec.rbd 75
5.1.39. .spec.rbd.secretRef 77
5.1.40. .spec.scaleIO 77
5.1.41. .spec.scaleIO.secretRef 78
5.1.42. .spec.storageos 79
5.1.43. .spec.storageos.secretRef 80
5.1.44. .spec.vsphereVolume 81
5.1.45. .status 82
5.2. API ENDPOINTS 83
5.2.1. /api/v1/persistentvolumes 84
5.2.2. /api/v1/watch/persistentvolumes 86
5.2.3. /api/v1/persistentvolumes/{name} 86
5.2.4. /api/v1/watch/persistentvolumes/{name} 89
5.2.5. /api/v1/persistentvolumes/{name}/status 90

CHAPTER 6. PERSISTENTVOLUMECLAIM [V1] 93
6.1. SPECIFICATION 93
6.1.1. .spec 94
6.1.2. .spec.dataSource 98

6.1.3. .spec.dataSourceRef 98
6.1.4. .spec.resources 99
6.1.5. .status 100
6.1.6. .status.conditions 103
6.1.7. .status.conditions[] 104
6.1.8. .status.modifyVolumeStatus 104
6.2. API ENDPOINTS 105
6.2.1. /api/v1/persistentvolumeclaims 106
6.2.2. /api/v1/watch/persistentvolumeclaims 107
6.2.3. /api/v1/namespaces/{namespace}/persistentvolumeclaims 107
6.2.4. /api/v1/watch/namespaces/{namespace}/persistentvolumeclaims 109
6.2.5. /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name} 109
6.2.6. /api/v1/watch/namespaces/{namespace}/persistentvolumeclaims/{name} 112
6.2.7. /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}/status 113

CHAPTER 7. STORAGECLASS [STORAGE.K8S.IO/V1] 116
7.1. SPECIFICATION 116
7.2. API ENDPOINTS 118
7.2.1. /apis/storage.k8s.io/v1/storageclasses 119
7.2.2. /apis/storage.k8s.io/v1/watch/storageclasses 121
7.2.3. /apis/storage.k8s.io/v1/storageclasses/{name} 121
7.2.4. /apis/storage.k8s.io/v1/watch/storageclasses/{name} 124

CHAPTER 8. STORAGESTATE [MIGRATION.K8S.IO/V1ALPHA1] 126
8.1. SPECIFICATION 126
8.1.1. .spec 126
8.1.2. .spec.resource 127
8.1.3. .status 127
8.2. API ENDPOINTS 128
8.2.1. /apis/migration.k8s.io/v1alpha1/storagestates 128
8.2.2. /apis/migration.k8s.io/v1alpha1/storagestates/{name} 130
8.2.3. /apis/migration.k8s.io/v1alpha1/storagestates/{name}/status 133

CHAPTER 9. STORAGEVERSIONMIGRATION [MIGRATION.K8S.IO/V1ALPHA1] 137
9.1. SPECIFICATION 137
9.1.1. .spec 137
9.1.2. .spec.resource 138
9.1.3. .status 138
9.1.4. .status.conditions 139
9.1.5. .status.conditions[] 139
9.2. API ENDPOINTS 139
9.2.1. /apis/migration.k8s.io/v1alpha1/storageversionmigrations 140
9.2.2. /apis/migration.k8s.io/v1alpha1/storageversionmigrations/{name} 142
9.2.3. /apis/migration.k8s.io/v1alpha1/storageversionmigrations/{name}/status 145

CHAPTER 10. VOLUMEATTACHMENT [STORAGE.K8S.IO/V1] 148
10.1. SPECIFICATION 148
10.1.1. .spec 149
10.1.2. .spec.source 149
10.1.3. .status 150
10.1.4. .status.attachError 151
10.1.5. .status.detachError 151
10.2. API ENDPOINTS 152
10.2.1. /apis/storage.k8s.io/v1/volumeattachments 153

10.2.2. /apis/storage.k8s.io/v1/watch/volumeattachments 155
10.2.3. /apis/storage.k8s.io/v1/volumeattachments/{name} 155
10.2.4. /apis/storage.k8s.io/v1/watch/volumeattachments/{name} 158
10.2.5. /apis/storage.k8s.io/v1/volumeattachments/{name}/status 159

CHAPTER 11. VOLUMESNAPSHOT [SNAPSHOT.STORAGE.K8S.IO/V1] 162
11.1. SPECIFICATION 162
11.1.1. .spec 163
11.1.2. .spec.source 164
11.1.3. .status 165
11.1.4. .status.error 167
11.2. API ENDPOINTS 168
11.2.1. /apis/snapshot.storage.k8s.io/v1/volumesnapshots 168
11.2.2. /apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots 169
11.2.3. /apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots/{name} 170
11.2.4. /apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots/{name}/status 173

CHAPTER 12. VOLUMESNAPSHOTCLASS [SNAPSHOT.STORAGE.K8S.IO/V1] 177
12.1. SPECIFICATION 177
12.2. API ENDPOINTS 178
12.2.1. /apis/snapshot.storage.k8s.io/v1/volumesnapshotclasses 179
12.2.2. /apis/snapshot.storage.k8s.io/v1/volumesnapshotclasses/{name} 180

CHAPTER 13. VOLUMESNAPSHOTCONTENT [SNAPSHOT.STORAGE.K8S.IO/V1] 184
13.1. SPECIFICATION 184
13.1.1. .spec 185
13.1.2. .spec.source 187
13.1.3. .spec.volumeSnapshotRef 188
13.1.4. .status 190
13.1.5. .status.error 192
13.2. API ENDPOINTS 192
13.2.1. /apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents 192
13.2.2. /apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents/{name} 194
13.2.3. /apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents/{name}/status 197

CHAPTER 1. STORAGE APIS

1.1. CSIDRIVER [STORAGE.K8S.IO/V1]

Description

CSIDriver captures information about a Container Storage Interface (CSI) volume driver deployed on the cluster. Kubernetes attach detach controller uses this object to determine whether attach is required. Kubelet uses this object to determine whether pod information needs to be passed on mount. CSIDriver objects are non-namespaced.

Type

object

1.2. CSINODE [STORAGE.K8S.IO/V1]

Description

CSINode holds information about all CSI drivers installed on a node. CSI drivers do not need to create the CSINode object directly. As long as they use the node-driver-registrar sidecar container, the kubelet will automatically populate the CSINode object for the CSI driver as part of kubelet plugin registration. CSINode has the same name as a node. If the object is missing, it means either there are no CSI Drivers available on the node, or the Kubelet version is low enough that it doesn't create this object. CSINode has an OwnerReference that points to the corresponding node object.

Type

object

1.3. CSISTORAGECAPACITY [STORAGE.K8S.IO/V1]

Description

CSIStorageCapacity stores the result of one CSI GetCapacity call. For a given StorageClass, this describes the available capacity in a particular topology segment. This can be used when considering where to instantiate new PersistentVolumes.
For example this can express things like: - StorageClass "standard" has "1234 GiB" available in "topology.kubernetes.io/zone=us-east1" - StorageClass "localssd" has "10 GiB" available in "kubernetes.io/hostname=knode-abc123"

The following three cases all imply that no capacity is available for a certain combination: - no object exists with suitable topology and storage class name - such an object exists, but the capacity is unset - such an object exists, but the capacity is zero

The producer of these objects can decide which approach is more suitable.

They are consumed by the kube-scheduler when a CSI driver opts into capacity-aware scheduling with CSIDriverSpec.StorageCapacity. The scheduler compares the MaximumVolumeSize against the requested size of pending volumes to filter out unsuitable nodes. If MaximumVolumeSize is unset, it falls back to a comparison against the less precise Capacity. If that is also unset, the scheduler assumes that capacity is insufficient and tries some other node.

Type

object

1.4. PERSISTENTVOLUME [V1]

Description

PersistentVolume (PV) is a storage resource provisioned by an administrator. It is analogous to a node. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes

Type

object

1.5. PERSISTENTVOLUMECLAIM [V1]

Description

PersistentVolumeClaim is a user's request for and claim to a persistent volume

Type

object

1.6. STORAGECLASS [STORAGE.K8S.IO/V1]

Description

StorageClass describes the parameters for a class of storage for which PersistentVolumes can be dynamically provisioned.
StorageClasses are non-namespaced; the name of the storage class according to etcd is in ObjectMeta.Name.

Type

object

1.7. STORAGESTATE [MIGRATION.K8S.IO/V1ALPHA1]

Description

The state of the storage of a specific resource.

Type

object

1.8. STORAGEVERSIONMIGRATION [MIGRATION.K8S.IO/V1ALPHA1]

Description

StorageVersionMigration represents a migration of stored data to the latest storage version.

Type

object

1.9. VOLUMEATTACHMENT [STORAGE.K8S.IO/V1]

Description

VolumeAttachment captures the intent to attach or detach the specified volume to/from the specified node.
VolumeAttachment objects are non-namespaced.

Type

object

1.10. VOLUMESNAPSHOT [SNAPSHOT.STORAGE.K8S.IO/V1]

Description

VolumeSnapshot is a user's request for either creating a point-in-time snapshot of a persistent volume, or binding to a pre-existing snapshot.

Type

object

1.11. VOLUMESNAPSHOTCLASS [SNAPSHOT.STORAGE.K8S.IO/V1]

Description

VolumeSnapshotClass specifies parameters that a underlying storage system uses when creating a volume snapshot. A specific VolumeSnapshotClass is used by specifying its name in a VolumeSnapshot object. VolumeSnapshotClasses are non-namespaced

Type

object

1.12. VOLUMESNAPSHOTCONTENT [SNAPSHOT.STORAGE.K8S.IO/V1]

Description

VolumeSnapshotContent represents the actual "on-disk" snapshot object in the underlying storage system

Type

object

CHAPTER 2. CSIDRIVER [STORAGE.K8S.IO/V1]

Description

CSIDriver captures information about a Container Storage Interface (CSI) volume driver deployed on the cluster. Kubernetes attach detach controller uses this object to determine whether attach is required. Kubelet uses this object to determine whether pod information needs to be passed on mount. CSIDriver objects are non-namespaced.

Type

object

Required

- **spec**

2.1. 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**	Standard object metadata. metadata.Name indicates the name of the CSI driver that this object refers to; it MUST be the same name returned by the CSI GetPluginName() call for that driver. The driver name must be 63 characters or less, beginning and ending with an alphanumeric character ([a-z0-9A-Z]) with dashes (-), dots (.), and alphanumerics between. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	CSIDriverSpec is the specification of a CSIDriver.

2.1.1. .spec

Description

CSIDriverSpec is the specification of a CSIDriver.

Type

object

Property	Type	Description

Property	Type	Description
attachRequired	**boolean**	attachRequired indicates this CSI volume driver requires an attach operation (because it implements the CSI ControllerPublishVolume() method), and that the Kubernetes attach detach controller should call the attach volume interface which checks the volumeattachment status and waits until the volume is attached before proceeding to mounting. The CSI external-attacher coordinates with CSI volume driver and updates the volumeattachment status when the attach operation is complete. If the CSIDriverRegistry feature gate is enabled and the value is specified to false, the attach operation will be skipped. Otherwise the attach operation will be called. This field is immutable.
fsGroupPolicy	**string**	fsGroupPolicy defines if the underlying volume supports changing ownership and permission of the volume before being mounted. Refer to the specific FSGroupPolicy values for additional details. This field was immutable in Kubernetes < 1.29 and now is mutable. Defaults to ReadWriteOnceWithFSType, which will examine each volume to determine if Kubernetes should modify ownership and permissions of the volume. With the default policy the defined fsGroup will only be applied if a fstype is defined and the volume's access mode contains ReadWriteOnce.
podInfoOnMount	**boolean**	podInfoOnMount indicates this CSI volume driver requires additional pod information (like

Property	Type	Description
		podName, podUID, etc.) during mount operations, if set to true. If set to false, pod information will not be passed on mount. Default is false. The CSI driver specifies podInfoOnMount as part of driver deployment. If true, Kubelet will pass pod information as VolumeContext in the CSI NodePublishVolume() calls. The CSI driver is responsible for parsing and validating the information passed in as VolumeContext. The following VolumeContext will be passed if podInfoOnMount is set to true. This list might grow, but the prefix will be used. "csi.storage.k8s.io/pod.name": pod.Name "csi.storage.k8s.io/pod.namespace": pod.Namespace "csi.storage.k8s.io/pod.uid": string(pod.UID) "csi.storage.k8s.io/ephemeral": "true" if the volume is an ephemeral inline volume defined by a CSIVolumeSource, otherwise "false" "csi.storage.k8s.io/ephemeral" is a new feature in Kubernetes 1.16. It is only required for drivers which support both the "Persistent" and "Ephemeral" VolumeLifecycleMode. Other drivers can leave pod info disabled and/or ignore this field. As Kubernetes 1.15 doesn't support this field, drivers can only support one mode when deployed on such a cluster and the deployment determines which mode that is, for example via a command line parameter of the driver. This field was immutable in Kubernetes < 1.29 and now is mutable.

Property	Type	Description
requiresRepublish	**boolean**	requiresRepublish indicates the CSI driver wants **NodePublishVolume** being periodically called to reflect any possible change in the mounted volume. This field defaults to false. Note: After a successful initial NodePublishVolume call, subsequent calls to NodePublishVolume should only update the contents of the volume. New mount points will not be seen by a running container.
seLinuxMount	**boolean**	seLinuxMount specifies if the CSI driver supports "-o context" mount option. When "true", the CSI driver must ensure that all volumes provided by this CSI driver can be mounted separately with different **-o context** options. This is typical for storage backends that provide volumes as filesystems on block devices or as independent shared volumes. Kubernetes will call NodeStage / NodePublish with "-o context=xyz" mount option when mounting a ReadWriteOncePod volume used in Pod that has explicitly set SELinux context. In the future, it may be expanded to other volume AccessModes. In any case, Kubernetes will ensure that the volume is mounted only with a single SELinux context. When "false", Kubernetes won't pass any special SELinux mount options to the driver. This is typical for volumes that represent subdirectories of a bigger shared filesystem. Default is "false".

Property	Type	Description
storageCapacity	**boolean**	storageCapacity indicates that the CSI volume driver wants pod scheduling to consider the storage capacity that the driver deployment will report by creating CSIStorageCapacity objects with capacity information, if set to true. The check can be enabled immediately when deploying a driver. In that case, provisioning new volumes with late binding will pause until the driver deployment has published some suitable CSIStorageCapacity object. Alternatively, the driver can be deployed with the field unset or false and it can be flipped later when storage capacity information has been published. This field was immutable in Kubernetes ⇐ 1.22 and now is mutable.
tokenRequests	**array**	tokenRequests indicates the CSI driver needs pods' service account tokens it is mounting volume for to do necessary authentication. Kubelet will pass the tokens in VolumeContext in the CSI NodePublishVolume calls. The CSI driver should parse and validate the following VolumeContext: "csi.storage.k8s.io/serviceAccount.tokens": { "<audience>": { "token": <token>, "expirationTimestamp": <expiration timestamp in RFC3339>, }, ... } Note: Audience in each TokenRequest should be different and at most one token is empty string. To receive a new token after expiry, RequiresRepublish can be used to trigger NodePublishVolume periodically.

Property	Type	Description
tokenRequests[]	**object**	TokenRequest contains parameters of a service account token.
volumeLifecycleModes	**array (string)**	volumeLifecycleModes defines what kind of volumes this CSI volume driver supports. The default if the list is empty is "Persistent", which is the usage defined by the CSI specification and implemented in Kubernetes via the usual PV/PVC mechanism. The other mode is "Ephemeral". In this mode, volumes are defined inline inside the pod spec with CSIVolumeSource and their lifecycle is tied to the lifecycle of that pod. A driver has to be aware of this because it is only going to get a NodePublishVolume call for such a volume. For more information about implementing this mode, see https://kubernetes-csi.github.io/docs/ephemeral-local-volumes.html A driver can support one or more of these modes and more modes may be added in the future. This field is beta. This field is immutable.

2.1.2. .spec.tokenRequests

Description

tokenRequests indicates the CSI driver needs pods' service account tokens it is mounting volume for to do necessary authentication. Kubelet will pass the tokens in VolumeContext in the CSI NodePublishVolume calls. The CSI driver should parse and validate the following VolumeContext: "csi.storage.k8s.io/serviceAccount.tokens": { "<audience>": { "token": <token>, "expirationTimestamp": <expiration timestamp in RFC3339>, }, ... }
Note: Audience in each TokenRequest should be different and at most one token is empty string. To receive a new token after expiry, RequiresRepublish can be used to trigger NodePublishVolume periodically.

Type

array

2.1.3. .spec.tokenRequests[]

Description

TokenRequest contains parameters of a service account token.

Type

object

Required

- **audience**

Property	Type	Description
audience	**string**	audience is the intended audience of the token in "TokenRequestSpec". It will default to the audiences of kube apiserver.
expirationSeconds	**integer**	expirationSeconds is the duration of validity of the token in "TokenRequestSpec". It has the same default value of "ExpirationSeconds" in "TokenRequestSpec".

2.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/storage.k8s.io/v1/csidrivers**
  - **DELETE**: delete collection of CSIDriver
  - **GET**: list or watch objects of kind CSIDriver
  - **POST**: create a CSIDriver
- **/apis/storage.k8s.io/v1/watch/csidrivers**
  - **GET**: watch individual changes to a list of CSIDriver. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/storage.k8s.io/v1/csidrivers/{name}**
  - **DELETE**: delete a CSIDriver
  - **GET**: read the specified CSIDriver
  - **PATCH**: partially update the specified CSIDriver
  - **PUT**: replace the specified CSIDriver

- **/apis/storage.k8s.io/v1/watch/csidrivers/{name}**
  - **GET**: watch changes to an object of kind CSIDriver. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

2.2.1. /apis/storage.k8s.io/v1/csidrivers

HTTP method

DELETE

Description

delete collection of CSIDriver

Table 2.1. 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

Table 2.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind CSIDriver

Table 2.3. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIDriverList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a CSIDriver

Table 2.4. 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.

Table 2.5. Body parameters

Parameter	Type	Description
body	**CSIDriver** schema	

Table 2.6. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIDriver** schema
201 - Created	**CSIDriver** schema
202 - Accepted	**CSIDriver** schema
401 - Unauthorized	Empty

2.2.2. /apis/storage.k8s.io/v1/watch/csidrivers

HTTP method

GET

Description

watch individual changes to a list of CSIDriver. deprecated: use the 'watch' parameter with a list operation instead.

Table 2.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

2.2.3. /apis/storage.k8s.io/v1/csidrivers/{name}

Table 2.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSIDriver

HTTP method

DELETE

Description

delete a CSIDriver

Table 2.9. 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

Table 2.10. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIDriver** schema
202 - Accepted	**CSIDriver** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified CSIDriver

Table 2.11. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIDriver** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified CSIDriver

Table 2.12. 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.

Table 2.13. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIDriver** schema
201 - Created	**CSIDriver** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified CSIDriver

Table 2.14. 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.

Table 2.15. Body parameters

Parameter	Type	Description
body	**CSIDriver** schema	

Table 2.16. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIDriver** schema
201 - Created	**CSIDriver** schema
401 - Unauthorized	Empty

2.2.4. /apis/storage.k8s.io/v1/watch/csidrivers/{name}

Table 2.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSIDriver

HTTP method

GET

Description

watch changes to an object of kind CSIDriver. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 2.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

CHAPTER 3. CSINODE [STORAGE.K8S.IO/V1]

Description

CSINode holds information about all CSI drivers installed on a node. CSI drivers do not need to create the CSINode object directly. As long as they use the node-driver-registrar sidecar container, the kubelet will automatically populate the CSINode object for the CSI driver as part of kubelet plugin registration. CSINode has the same name as a node. If the object is missing, it means either there are no CSI Drivers available on the node, or the Kubelet version is low enough that it doesn't create this object. CSINode has an OwnerReference that points to the corresponding node object.

Type

object

Required

- **spec**

3.1. 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
metadata	**ObjectMeta**	Standard object's metadata. metadata.name must be the Kubernetes node name.
spec	**object**	CSINodeSpec holds information about the specification of all CSI drivers installed on a node

3.1.1. .spec

Description

CSINodeSpec holds information about the specification of all CSI drivers installed on a node

Type

object

Required

- **drivers**

Property	Type	Description
drivers	**array**	drivers is a list of information of all CSI Drivers existing on a node. If all drivers in the list are uninstalled, this can become empty.
drivers[]	**object**	CSINodeDriver holds information about the specification of one CSI driver installed on a node

3.1.2. .spec.drivers

Description

drivers is a list of information of all CSI Drivers existing on a node. If all drivers in the list are uninstalled, this can become empty.

Type

array

3.1.3. .spec.drivers[]

Description

CSINodeDriver holds information about the specification of one CSI driver installed on a node

Type

object

Required

- **name**
- **nodeID**

Property	Type	Description

Property	Type	Description
allocatable	**object**	VolumeNodeResources is a set of resource limits for scheduling of volumes.
name	**string**	name represents the name of the CSI driver that this object refers to. This MUST be the same name returned by the CSI GetPluginName() call for that driver.
nodeID	**string**	nodeID of the node from the driver point of view. This field enables Kubernetes to communicate with storage systems that do not share the same nomenclature for nodes. For example, Kubernetes may refer to a given node as "node1", but the storage system may refer to the same node as "nodeA". When Kubernetes issues a command to the storage system to attach a volume to a specific node, it can use this field to refer to the node name using the ID that the storage system will understand, e.g. "nodeA" instead of "node1". This field is required.
topologyKeys	**array (string)**	topologyKeys is the list of keys supported by the driver. When a driver is initialized on a cluster, it provides a set of topology keys that it understands (e.g. "company.com/zone", "company.com/region"). When a driver is initialized on a node, it provides the same topology keys along with values. Kubelet will expose these topology keys as labels on its own node object. When Kubernetes does topology aware provisioning, it can use this list to determine which labels it should retrieve from the node object and pass back to the driver. It is possible for different nodes to use different topology keys. This can be empty if driver does not support topology.

3.1.4. .spec.drivers[].allocatable

Description

VolumeNodeResources is a set of resource limits for scheduling of volumes.

Type

object

Property	Type	Description
count	**integer**	count indicates the maximum number of unique volumes managed by the CSI driver that can be used on a node. A volume that is both attached and mounted on a node is considered to be used once, not twice. The same rule applies for a unique volume that is shared among multiple pods on the same node. If this field is not specified, then the supported number of volumes on this node is unbounded.

3.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/storage.k8s.io/v1/csinodes**
  - **DELETE**: delete collection of CSINode
  - **GET**: list or watch objects of kind CSINode
  - **POST**: create a CSINode
- **/apis/storage.k8s.io/v1/watch/csinodes**
  - **GET**: watch individual changes to a list of CSINode. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/storage.k8s.io/v1/csinodes/{name}**
  - **DELETE**: delete a CSINode
  - **GET**: read the specified CSINode
  - **PATCH**: partially update the specified CSINode
  - **PUT**: replace the specified CSINode
- **/apis/storage.k8s.io/v1/watch/csinodes/{name}**

- **GET**: watch changes to an object of kind CSINode. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

3.2.1. /apis/storage.k8s.io/v1/csinodes

HTTP method

DELETE

Description

delete collection of CSINode

Table 3.1. 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

Table 3.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind CSINode

Table 3.3. HTTP responses

HTTP code	Reponse body
200 - OK	**CSINodeList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a CSINode

Table 3.4. 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.

Table 3.5. Body parameters

Parameter	Type	Description
body	**CSINode** schema	

Table 3.6. HTTP responses

HTTP code	Reponse body
200 - OK	**CSINode** schema
201 - Created	**CSINode** schema
202 - Accepted	**CSINode** schema
401 - Unauthorized	Empty

3.2.2. /apis/storage.k8s.io/v1/watch/csinodes

HTTP method

GET

Description

watch individual changes to a list of CSINode. deprecated: use the 'watch' parameter with a list operation instead.

Table 3.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

3.2.3. /apis/storage.k8s.io/v1/csinodes/{name}

Table 3.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSINode

HTTP method

DELETE

Description

delete a CSINode

Table 3.9. 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

Table 3.10. HTTP responses

HTTP code	Reponse body
200 - OK	**CSINode** schema
202 - Accepted	**CSINode** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified CSINode

Table 3.11. HTTP responses

HTTP code	Reponse body
200 - OK	**CSINode** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified CSINode

Table 3.12. 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.

Table 3.13. HTTP responses

HTTP code	Reponse body
200 - OK	**CSINode** schema
201 - Created	**CSINode** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified CSINode

Table 3.14. 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.

Table 3.15. Body parameters

Parameter	Type	Description
body	**CSINode** schema	

Table 3.16. HTTP responses

HTTP code	Reponse body
200 - OK	**CSINode** schema
201 - Created	**CSINode** schema
401 - Unauthorized	Empty

3.2.4. /apis/storage.k8s.io/v1/watch/csinodes/{name}

Table 3.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSINode

HTTP method

GET

Description

watch changes to an object of kind CSINode. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 3.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

CHAPTER 4. CSISTORAGECAPACITY [STORAGE.K8S.IO/V1]

Description

CSIStorageCapacity stores the result of one CSI GetCapacity call. For a given StorageClass, this describes the available capacity in a particular topology segment. This can be used when considering where to instantiate new PersistentVolumes.
For example this can express things like: - StorageClass "standard" has "1234 GiB" available in "topology.kubernetes.io/zone=us-east1" - StorageClass "localssd" has "10 GiB" available in "kubernetes.io/hostname=knode-abc123"

The following three cases all imply that no capacity is available for a certain combination: - no object exists with suitable topology and storage class name - such an object exists, but the capacity is unset - such an object exists, but the capacity is zero

The producer of these objects can decide which approach is more suitable.

They are consumed by the kube-scheduler when a CSI driver opts into capacity-aware scheduling with CSIDriverSpec.StorageCapacity. The scheduler compares the MaximumVolumeSize against the requested size of pending volumes to filter out unsuitable nodes. If MaximumVolumeSize is unset, it falls back to a comparison against the less precise Capacity. If that is also unset, the scheduler assumes that capacity is insufficient and tries some other node.

Type

object

Required

- **storageClassName**

4.1. 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
capacity	**Quantity**	capacity is the value reported by the CSI driver in its GetCapacityResponse for a GetCapacityRequest with topology and parameters that match the previous fields. The semantic is currently (CSI spec 1.2) defined as: The available capacity, in bytes, of the storage that can be used to provision volumes. If not set, that information is currently unavailable.
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
maximumVolumeSize	**Quantity**	maximumVolumeSize is the value reported by the CSI driver in its GetCapacityResponse for a GetCapacityRequest with topology and parameters that match the previous fields. This is defined since CSI spec 1.4.0 as the largest size that may be used in a CreateVolumeRequest.capacity_range.required_bytes field to create a volume with the same parameters as those in GetCapacityRequest. The corresponding value in the Kubernetes API is ResourceRequirements.Requests in a volume claim.

Property	Type	Description
metadata	**ObjectMeta**	Standard object's metadata. The name has no particular meaning. It must be a DNS subdomain (dots allowed, 253 characters). To ensure that there are no conflicts with other CSI drivers on the cluster, the recommendation is to use csisc-<uuid>, a generated name, or a reverse-domain name which ends with the unique CSI driver name. Objects are namespaced. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
nodeTopology	**LabelSelector**	nodeTopology defines which nodes have access to the storage for which capacity was reported. If not set, the storage is not accessible from any node in the cluster. If empty, the storage is accessible from all nodes. This field is immutable.
storageClassName	**string**	storageClassName represents the name of the StorageClass that the reported capacity applies to. It must meet the same requirements as the name of a StorageClass object (non-empty, DNS subdomain). If that object no longer exists, the CSIStorageCapacity object is obsolete and should be removed by its creator. This field is immutable.

4.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/storage.k8s.io/v1/csistoragecapacities**
  - **GET**: list or watch objects of kind CSIStorageCapacity

- **/apis/storage.k8s.io/v1/watch/csistoragecapacities**
  - **GET**: watch individual changes to a list of CSIStorageCapacity. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/storage.k8s.io/v1/namespaces/{namespace}/csistoragecapacities**
  - **DELETE**: delete collection of CSIStorageCapacity
  - **GET**: list or watch objects of kind CSIStorageCapacity
  - **POST**: create a CSIStorageCapacity
- **/apis/storage.k8s.io/v1/watch/namespaces/{namespace}/csistoragecapacities**
  - **GET**: watch individual changes to a list of CSIStorageCapacity. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/storage.k8s.io/v1/namespaces/{namespace}/csistoragecapacities/{name}**
  - **DELETE**: delete a CSIStorageCapacity
  - **GET**: read the specified CSIStorageCapacity
  - **PATCH**: partially update the specified CSIStorageCapacity
  - **PUT**: replace the specified CSIStorageCapacity
- **/apis/storage.k8s.io/v1/watch/namespaces/{namespace}/csistoragecapacities/{name}**
  - **GET**: watch changes to an object of kind CSIStorageCapacity. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

4.2.1. /apis/storage.k8s.io/v1/csistoragecapacities

HTTP method

GET

Description

list or watch objects of kind CSIStorageCapacity

Table 4.1. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIStorageCapacityList** schema
401 - Unauthorized	Empty

4.2.2. /apis/storage.k8s.io/v1/watch/csistoragecapacities

HTTP method

GET

Description

watch individual changes to a list of CSIStorageCapacity. deprecated: use the 'watch' parameter with a list operation instead.

Table 4.2. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

4.2.3. /apis/storage.k8s.io/v1/namespaces/{namespace}/csistoragecapacities

HTTP method

DELETE

Description

delete collection of CSIStorageCapacity

Table 4.3. 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

Table 4.4. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind CSIStorageCapacity

Table 4.5. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIStorageCapacityList** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

POST

Description

create a CSIStorageCapacity

Table 4.6. 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.

Table 4.7. Body parameters

Parameter	Type	Description
body	**CSIStorageCapacity** schema	

Table 4.8. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIStorageCapacity** schema
201 - Created	**CSIStorageCapacity** schema
202 - Accepted	**CSIStorageCapacity** schema
401 - Unauthorized	Empty

4.2.4. /apis/storage.k8s.io/v1/watch/namespaces/{namespace}/csistoragecapacities

HTTP method

GET

Description

watch individual changes to a list of CSIStorageCapacity. deprecated: use the 'watch' parameter with a list operation instead.

Table 4.9. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

4.2.5. /apis/storage.k8s.io/v1/namespaces/{namespace}/csistoragecapacities/{name

Table 4.10. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSIStorageCapacity

HTTP method

DELETE

Description

delete a CSIStorageCapacity

Table 4.11. Query parameters

Parameter	Type	Description

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

Table 4.12. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified CSIStorageCapacity

Table 4.13. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIStorageCapacity** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified CSIStorageCapacity

Table 4.14. 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.

Table 4.15. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIStorageCapacity** schema
201 - Created	**CSIStorageCapacity** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified CSIStorageCapacity

Table 4.16. 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.

Table 4.17. Body parameters

Parameter	Type	Description
body	**CSIStorageCapacity** schema	

Table 4.18. HTTP responses

HTTP code	Reponse body
200 - OK	**CSIStorageCapacity** schema
201 - Created	**CSIStorageCapacity** schema
401 - Unauthorized	Empty

4.2.6. /apis/storage.k8s.io/v1/watch/namespaces/{namespace}/csistoragecapacities

Table 4.19. Global path parameters

Parameter	Type	Description
name	**string**	name of the CSIStorageCapacity

HTTP method

GET

Description

watch changes to an object of kind CSIStorageCapacity. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 4.20. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

CHAPTER 5. PERSISTENTVOLUME [V1]

Description

PersistentVolume (PV) is a storage resource provisioned by an administrator. It is analogous to a node. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes

Type

object

5.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	PersistentVolumeSpec is the specification of a persistent volume.
status	**object**	PersistentVolumeStatus is the current status of a persistent volume.

5.1.1. .spec

Description

PersistentVolumeSpec is the specification of a persistent volume.

Type

object

Property	Type	Description
accessModes	**array (string)**	accessModes contains all ways the volume can be mounted. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes
awsElasticBlockStore	**object**	Represents a Persistent Disk resource in AWS. An AWS EBS disk must exist before mounting to a container. The disk must also be in the same AWS zone as the kubelet. An AWS EBS disk can only be mounted as read/write once. AWS EBS volumes support ownership management and SELinux relabeling.
azureDisk	**object**	AzureDisk represents an Azure Data Disk mount on the host and bind mount to the pod.
azureFile	**object**	AzureFile represents an Azure File Service mount on the host and bind mount to the pod.
capacity	**object (Quantity)**	capacity is the description of the persistent volume's resources and capacity. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#capacity
cephfs	**object**	Represents a Ceph Filesystem mount that lasts the lifetime of a pod Cephfs volumes do not support ownership management or SELinux relabeling.

Property	Type	Description
cinder	**object**	Represents a cinder volume resource in Openstack. A Cinder volume must exist before mounting to a container. The volume must also be in the same region as the kubelet. Cinder volumes support ownership management and SELinux relabeling.
claimRef	**object**	ObjectReference contains enough information to let you inspect or modify the referred object.
csi	**object**	Represents storage that is managed by an external CSI volume driver (Beta feature)
fc	**object**	Represents a Fibre Channel volume. Fibre Channel volumes can only be mounted as read/write once. Fibre Channel volumes support ownership management and SELinux relabeling.
flexVolume	**object**	FlexPersistentVolumeSource represents a generic persistent volume resource that is provisioned/attached using an exec based plugin.
flocker	**object**	Represents a Flocker volume mounted by the Flocker agent. One and only one of datasetName and datasetUUID should be set. Flocker volumes do not support ownership management or SELinux relabeling.

Property	Type	Description
gcePersistentDisk	**object**	Represents a Persistent Disk resource in Google Compute Engine. A GCE PD must exist before mounting to a container. The disk must also be in the same GCE project and zone as the kubelet. A GCE PD can only be mounted as read/write once or read-only many times. GCE PDs support ownership management and SELinux relabeling.
glusterfs	**object**	Represents a Glusterfs mount that lasts the lifetime of a pod. Glusterfs volumes do not support ownership management or SELinux relabeling.
hostPath	**object**	Represents a host path mapped into a pod. Host path volumes do not support ownership management or SELinux relabeling.
iscsi	**object**	ISCSIPersistentVolumeSource represents an ISCSI disk. ISCSI volumes can only be mounted as read/write once. ISCSI volumes support ownership management and SELinux relabeling.
local	**object**	Local represents directly-attached storage with node affinity (Beta feature)
mountOptions	**array (string)**	mountOptions is the list of mount options, e.g. ["ro", "soft"]. Not validated - mount will simply fail if one is invalid. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes/#mount-options

Property	Type	Description
nfs	**object**	Represents an NFS mount that lasts the lifetime of a pod. NFS volumes do not support ownership management or SELinux relabeling.
nodeAffinity	**object**	VolumeNodeAffinity defines constraints that limit what nodes this volume can be accessed from.
persistentVolumeReclaimPolicy	**string**	persistentVolumeReclaimPolicy defines what happens to a persistent volume when released from its claim. Valid options are Retain (default for manually created PersistentVolumes), Delete (default for dynamically provisioned PersistentVolumes), and Recycle (deprecated). Recycle must be supported by the volume plugin underlying this PersistentVolume. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#reclaiming Possible enum values: - **"Delete"** means the volume will be deleted from Kubernetes on release from its claim. The volume plugin must support Deletion. - **"Recycle"** means the volume will be recycled back into the pool of unbound persistent volumes on release from its claim. The volume plugin must support Recycling. - **"Retain"** means the volume will be left in its current phase (Released) for manual reclamation by the administrator. The default policy is Retain.
photonPersistentDisk	**object**	Represents a Photon Controller persistent disk resource.
portworxVolume	**object**	PortworxVolumeSource represents a Portworx volume resource.

Property	Type	Description
quobyte	**object**	Represents a Quobyte mount that lasts the lifetime of a pod. Quobyte volumes do not support ownership management or SELinux relabeling.
rbd	**object**	Represents a Rados Block Device mount that lasts the lifetime of a pod. RBD volumes support ownership management and SELinux relabeling.
scaleIO	**object**	ScaleIOPersistentVolumeSource represents a persistent ScaleIO volume
storageClassName	**string**	storageClassName is the name of StorageClass to which this persistent volume belongs. Empty value means that this volume does not belong to any StorageClass.
storageos	**object**	Represents a StorageOS persistent volume resource.
volumeAttributesClassName	**string**	Name of VolumeAttributesClass to which this persistent volume belongs. Empty value is not allowed. When this field is not set, it indicates that this volume does not belong to any VolumeAttributesClass. This field is mutable and can be changed by the CSI driver after a volume has been updated successfully to a new class. For an unbound PersistentVolume, the volumeAttributesClassName will be matched with unbound PersistentVolumeClaims during the binding process. This is a beta field and requires enabling VolumeAttributesClass feature (off by default).

Property	Type	Description
volumeMode	**string**	volumeMode defines if a volume is intended to be used with a formatted filesystem or to remain in raw block state. Value of Filesystem is implied when not included in spec. Possible enum values: - **"Block"** means the volume will not be formatted with a filesystem and will remain a raw block device. - **"Filesystem"** means the volume will be or is formatted with a filesystem.
vsphereVolume	**object**	Represents a vSphere volume resource.

5.1.2. .spec.awsElasticBlockStore

Description

Represents a Persistent Disk resource in AWS.
An AWS EBS disk must exist before mounting to a container. The disk must also be in the same AWS zone as the kubelet. An AWS EBS disk can only be mounted as read/write once. AWS EBS volumes support ownership management and SELinux relabeling.

Type

object

Required

- **volumeID**

Property	Type	Description
fsType	**string**	fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore

Property	Type	Description
partition	**integer**	partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty).
readOnly	**boolean**	readOnly value true will force the readOnly setting in VolumeMounts. More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore
volumeID	**string**	volumeID is unique ID of the persistent disk resource in AWS (Amazon EBS volume). More info: https://kubernetes.io/docs/concepts/storage/volumes#awselasticblockstore

5.1.3. .spec.azureDisk

Description

AzureDisk represents an Azure Data Disk mount on the host and bind mount to the pod.

Type

object

Required

- **diskName**
- **diskURI**

Property	Type	Description
cachingMode	**string**	cachingMode is the Host Caching mode: None, Read Only, Read Write. Possible enum values: - **"None"** - **"ReadOnly"** - **"ReadWrite"**

Property	Type	Description
diskName	**string**	diskName is the Name of the data disk in the blob storage
diskURI	**string**	diskURI is the URI of data disk in the blob storage
fsType	**string**	fsType is Filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
kind	**string**	kind expected values are Shared: multiple blob disks per storage account Dedicated: single blob disk per storage account Managed: azure managed data disk (only in managed availability set). defaults to shared Possible enum values: - **"Dedicated"** - **"Managed"** - **"Shared"**
readOnly	**boolean**	readOnly Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.

5.1.4. .spec.azureFile

Description

AzureFile represents an Azure File Service mount on the host and bind mount to the pod.

Type

object

Required

- **secretName**
- **shareName**

Property	Type	Description

Property	Type	Description
readOnly	**boolean**	readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
secretName	**string**	secretName is the name of secret that contains Azure Storage Account Name and Key
secretNamespace	**string**	secretNamespace is the namespace of the secret that contains Azure Storage Account Name and Key default is the same as the Pod
shareName	**string**	shareName is the azure Share Name

5.1.5. .spec.cephfs

Description

Represents a Ceph Filesystem mount that lasts the lifetime of a pod Cephfs volumes do not support ownership management or SELinux relabeling.

Type

object

Required

- **monitors**

Property	Type	Description
monitors	**array (string)**	monitors is Required: Monitors is a collection of Ceph monitors More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it
path	**string**	path is Optional: Used as the mounted root, rather than the full Ceph tree, default is /

Property	Type	Description
readOnly	**boolean**	readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it
secretFile	**string**	secretFile is Optional: SecretFile is the path to key ring for User, default is /etc/ceph/user.secret More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it
secretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
user	**string**	user is Optional: User is the rados user name, default is admin More info: https://examples.k8s.io/volumes/cephfs/README.md#how-to-use-it

5.1.6. .spec.cephfs.secretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.7. .spec.cinder

Description

Represents a cinder volume resource in Openstack. A Cinder volume must exist before mounting to a container. The volume must also be in the same region as the kubelet. Cinder volumes support ownership management and SELinux relabeling.

Type

object

Required

- **volumeID**

Property	Type	Description
fsType	**string**	fsType Filesystem type to mount. Must be a filesystem type supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://examples.k8s.io/mysql-cinder-pd/README.md
readOnly	**boolean**	readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts. More info: https://examples.k8s.io/mysql-cinder-pd/README.md
secretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
volumeID	**string**	volumeID used to identify the volume in cinder. More info: https://examples.k8s.io/mysql-cinder-pd/README.md

5.1.8. .spec.cinder.secretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.9. .spec.claimRef

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

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.
kind	**string**	Kind of the referent. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds

Property	Type	Description
name	**string**	Name of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#names
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

5.1.10. .spec.csi

Description

Represents storage that is managed by an external CSI volume driver (Beta feature)

Type

object

Required

- **driver**
- **volumeHandle**

Property	Type	Description
controllerExpandSecretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Property	Type	Description
controllerPublishSecretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
driver	**string**	driver is the name of the driver to use for this volume. Required.
fsType	**string**	fsType to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs".
nodeExpandSecretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
nodePublishSecretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
nodeStageSecretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
readOnly	**boolean**	readOnly value to pass to ControllerPublishVolumeRequest. Defaults to false (read/write).
volumeAttributes	**object (string)**	volumeAttributes of the volume to publish.
volumeHandle	**string**	volumeHandle is the unique volume name returned by the CSI volume plugin's CreateVolume to refer to the volume on all subsequent calls. Required.

5.1.11. .spec.csi.controllerExpandSecretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.12. .spec.csi.controllerPublishSecretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.13. .spec.csi.nodeExpandSecretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.14. .spec.csi.nodePublishSecretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.15. .spec.csi.nodeStageSecretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.16. .spec.fc

Description

Represents a Fibre Channel volume. Fibre Channel volumes can only be mounted as read/write once. Fibre Channel volumes support ownership management and SELinux relabeling.

Type

object

Property	Type	Description
fsType	**string**	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
lun	**integer**	lun is Optional: FC target lun number
readOnly	**boolean**	readOnly is Optional: Defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
targetWWNs	**array (string)**	targetWWNs is Optional: FC target worldwide names (WWNs)
wwids	**array (string)**	wwids Optional: FC volume world wide identifiers (wwids) Either wwids or combination of targetWWNs and lun must be set, but not both simultaneously.

5.1.17. .spec.flexVolume

Description

FlexPersistentVolumeSource represents a generic persistent volume resource that is provisioned/attached using an exec based plugin.

Type

object

Required

- **driver**

Property	Type	Description
driver	**string**	driver is the name of the driver to use for this volume.

Property	Type	Description
fsType	**string**	fsType is the Filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". The default filesystem depends on FlexVolume script.
options	**object (string)**	options is Optional: this field holds extra command options if any.
readOnly	**boolean**	readOnly is Optional: defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
secretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

5.1.18. .spec.flexVolume.secretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.19. .spec.flocker

Description

Represents a Flocker volume mounted by the Flocker agent. One and only one of datasetName and datasetUUID should be set. Flocker volumes do not support ownership management or SELinux relabeling.

Type

object

Property	Type	Description
datasetName	**string**	datasetName is Name of the dataset stored as metadata → name on the dataset for Flocker should be considered as deprecated
datasetUUID	**string**	datasetUUID is the UUID of the dataset. This is unique identifier of a Flocker dataset

5.1.20. .spec.gcePersistentDisk

Description

Represents a Persistent Disk resource in Google Compute Engine.
A GCE PD must exist before mounting to a container. The disk must also be in the same GCE project and zone as the kubelet. A GCE PD can only be mounted as read/write once or read-only many times. GCE PDs support ownership management and SELinux relabeling.

Type

object

Required

- **pdName**

Property	Type	Description
fsType	**string**	fsType is filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk

Property	Type	Description
partition	**integer**	partition is the partition in the volume that you want to mount. If omitted, the default is to mount by volume name. Examples: For volume /dev/sda1, you specify the partition as "1". Similarly, the volume partition for /dev/sda is "0" (or you can leave the property empty). More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk
pdName	**string**	pdName is unique name of the PD resource in GCE. Used to identify the disk in GCE. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk
readOnly	**boolean**	readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#gcepersistentdisk

5.1.21. .spec.glusterfs

Description

Represents a Glusterfs mount that lasts the lifetime of a pod. Glusterfs volumes do not support ownership management or SELinux relabeling.

Type

object

Required

- **endpoints**
- **path**

Property	Type	Description

Property	Type	Description
endpoints	**string**	endpoints is the endpoint name that details Glusterfs topology. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod
endpointsNamespace	**string**	endpointsNamespace is the namespace that contains Glusterfs endpoint. If this field is empty, the EndpointNamespace defaults to the same namespace as the bound PVC. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod
path	**string**	path is the Glusterfs volume path. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod
readOnly	**boolean**	readOnly here will force the Glusterfs volume to be mounted with read-only permissions. Defaults to false. More info: https://examples.k8s.io/volumes/glusterfs/README.md#create-a-pod

5.1.22. .spec.hostPath

Description

Represents a host path mapped into a pod. Host path volumes do not support ownership management or SELinux relabeling.

Type

object

Required

- **path**

Property	Type	Description

Property	Type	Description
path	**string**	path of the directory on the host. If the path is a symlink, it will follow the link to the real path. More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath
type	**string**	type for HostPath Volume Defaults to "" More info: https://kubernetes.io/docs/concepts/storage/volumes#hostpath Possible enum values: - **""** For backwards compatible, leave it empty if unset - **"BlockDevice"** A block device must exist at the given path - **"CharDevice"** A character device must exist at the given path - **"Directory"** A directory must exist at the given path - **"DirectoryOrCreate"** If nothing exists at the given path, an empty directory will be created there as needed with file mode 0755, having the same group and ownership with Kubelet. - **"File"** A file must exist at the given path - **"FileOrCreate"** If nothing exists at the given path, an empty file will be created there as needed with file mode 0644, having the same group and ownership with Kubelet. - **"Socket"** A UNIX socket must exist at the given path

5.1.23. .spec.iscsi

Description

ISCSIPersistentVolumeSource represents an ISCSI disk. ISCSI volumes can only be mounted as read/write once. ISCSI volumes support ownership management and SELinux relabeling.

Type

object

Required

- **targetPortal**
- **iqn**
- **lun**

Property	Type	Description
chapAuthDiscovery	**boolean**	chapAuthDiscovery defines whether support iSCSI Discovery CHAP authentication
chapAuthSession	**boolean**	chapAuthSession defines whether support iSCSI Session CHAP authentication
fsType	**string**	fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#iscsi
initiatorName	**string**	initiatorName is the custom iSCSI Initiator Name. If initiatorName is specified with iscsiInterface simultaneously, new iSCSI interface <target portal>:<volume name> will be created for the connection.
iqn	**string**	iqn is Target iSCSI Qualified Name.
iscsiInterface	**string**	iscsiInterface is the interface Name that uses an iSCSI transport. Defaults to 'default' (tcp).
lun	**integer**	lun is iSCSI Target Lun number.
portals	**array (string)**	portals is the iSCSI Target Portal List. The Portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).
readOnly	**boolean**	readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false.

Property	Type	Description
secretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
targetPortal	**string**	targetPortal is iSCSI Target Portal. The Portal is either an IP or ip_addr:port if the port is other than default (typically TCP ports 860 and 3260).

5.1.24. .spec.iscsi.secretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.25. .spec.local

Description

Local represents directly-attached storage with node affinity (Beta feature)

Type

object

Required

- **path**

Property	Type	Description

Property	Type	Description
fsType	**string**	fsType is the filesystem type to mount. It applies only when the Path is a block device. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". The default value is to auto-select a filesystem if unspecified.
path	**string**	path of the full path to the volume on the node. It can be either a directory or block device (disk, partition, ...).

5.1.26. .spec.nfs

Description

Represents an NFS mount that lasts the lifetime of a pod. NFS volumes do not support ownership management or SELinux relabeling.

Type

object

Required

- **server**
- **path**

Property	Type	Description
path	**string**	path that is exported by the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs
readOnly	**boolean**	readOnly here will force the NFS export to be mounted with read-only permissions. Defaults to false. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs
server	**string**	server is the hostname or IP address of the NFS server. More info: https://kubernetes.io/docs/concepts/storage/volumes#nfs

5.1.27. .spec.nodeAffinity

Description

VolumeNodeAffinity defines constraints that limit what nodes this volume can be accessed from.

Type

object

Property	Type	Description
required	**object**	A node selector represents the union of the results of one or more label queries over a set of nodes; that is, it represents the OR of the selectors represented by the node selector terms.

5.1.28. .spec.nodeAffinity.required

Description

A node selector represents the union of the results of one or more label queries over a set of nodes; that is, it represents the OR of the selectors represented by the node selector terms.

Type

object

Required

- **nodeSelectorTerms**

Property	Type	Description
nodeSelectorTerms	**array**	Required. A list of node selector terms. The terms are ORed.
nodeSelectorTerms[]	**object**	A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

5.1.29. .spec.nodeAffinity.required.nodeSelectorTerms

Description

Required. A list of node selector terms. The terms are ORed.

Type

array

5.1.30. .spec.nodeAffinity.required.nodeSelectorTerms[]

Description

A null or empty node selector term matches no objects. The requirements of them are ANDed. The TopologySelectorTerm type implements a subset of the NodeSelectorTerm.

Type

object

Property	Type	Description
matchExpressions	**array**	A list of node selector requirements by node's labels.
matchExpressions[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.
matchFields	**array**	A list of node selector requirements by node's fields.
matchFields[]	**object**	A node selector requirement is a selector that contains values, a key, and an operator that relates the key and values.

5.1.31. .spec.nodeAffinity.required.nodeSelectorTerms[].matchExpressions

Description

A list of node selector requirements by node's labels.

Type

array

5.1.32. .spec.nodeAffinity.required.nodeSelectorTerms[].matchExpressions[]

Description

A node 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**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt. Possible enum values: - **"DoesNotExist"** - **"Exists"** - **"Gt"** - **"In"** - **"Lt"** - **"NotIn"**
values	**array (string)**	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. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

5.1.33. .spec.nodeAffinity.required.nodeSelectorTerms[].matchFields

Description

A list of node selector requirements by node's fields.

Type

array

5.1.34. .spec.nodeAffinity.required.nodeSelectorTerms[].matchFields[]

Description

A node 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**	The label key that the selector applies to.
operator	**string**	Represents a key's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Gt, and Lt. Possible enum values: - **"DoesNotExist"** - **"Exists"** - **"Gt"** - **"In"** - **"Lt"** - **"NotIn"**
values	**array (string)**	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. If the operator is Gt or Lt, the values array must have a single element, which will be interpreted as an integer. This array is replaced during a strategic merge patch.

5.1.35. .spec.photonPersistentDisk

Description

Represents a Photon Controller persistent disk resource.

Type

object

Required

- **pdID**

Property	Type	Description
fsType	**string**	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
pdID	**string**	pdID is the ID that identifies Photon Controller persistent disk

5.1.36. .spec.portworxVolume

Description

PortworxVolumeSource represents a Portworx volume resource.

Type

object

Required

- **volumeID**

Property	Type	Description
fsType	**string**	fSType represents the filesystem type to mount Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs". Implicitly inferred to be "ext4" if unspecified.
readOnly	**boolean**	readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
volumeID	**string**	volumeID uniquely identifies a Portworx volume

5.1.37. .spec.quobyte

Description

Represents a Quobyte mount that lasts the lifetime of a pod. Quobyte volumes do not support ownership management or SELinux relabeling.

Type

object

Required

- **registry**
- **volume**

Property	Type	Description
group	**string**	group to map volume access to Default is no group

Property	Type	Description
readOnly	**boolean**	readOnly here will force the Quobyte volume to be mounted with read-only permissions. Defaults to false.
registry	**string**	registry represents a single or multiple Quobyte Registry services specified as a string as host:port pair (multiple entries are separated with commas) which acts as the central registry for volumes
tenant	**string**	tenant owning the given Quobyte volume in the Backend Used with dynamically provisioned Quobyte volumes, value is set by the plugin
user	**string**	user to map volume access to Defaults to serivceaccount user
volume	**string**	volume is a string that references an already created Quobyte volume by name.

5.1.38. .spec.rbd

Description

Represents a Rados Block Device mount that lasts the lifetime of a pod. RBD volumes support ownership management and SELinux relabeling.

Type

object

Required

- **monitors**
- **image**

Property	Type	Description

Property	Type	Description
fsType	**string**	fsType is the filesystem type of the volume that you want to mount. Tip: Ensure that the filesystem type is supported by the host operating system. Examples: "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified. More info: https://kubernetes.io/docs/concepts/storage/volumes#rbd
image	**string**	image is the rados image name. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
keyring	**string**	keyring is the path to key ring for RBDUser. Default is /etc/ceph/keyring. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
monitors	**array (string)**	monitors is a collection of Ceph monitors. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
pool	**string**	pool is the rados pool name. Default is rbd. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
readOnly	**boolean**	readOnly here will force the ReadOnly setting in VolumeMounts. Defaults to false. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it
secretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
user	**string**	user is the rados user name. Default is admin. More info: https://examples.k8s.io/volumes/rbd/README.md#how-to-use-it

5.1.39. .spec.rbd.secretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.40. .spec.scaleIO

Description

ScaleIOPersistentVolumeSource represents a persistent ScaleIO volume

Type

object

Required

- **gateway**
- **system**
- **secretRef**

Property	Type	Description
fsType	**string**	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Default is "xfs"
gateway	**string**	gateway is the host address of the ScaleIO API Gateway.
protectionDomain	**string**	protectionDomain is the name of the ScaleIO Protection Domain for the configured storage.

Property	Type	Description
readOnly	**boolean**	readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
secretRef	**object**	SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace
sslEnabled	**boolean**	sslEnabled is the flag to enable/disable SSL communication with Gateway, default false
storageMode	**string**	storageMode indicates whether the storage for a volume should be ThickProvisioned or ThinProvisioned. Default is ThinProvisioned.
storagePool	**string**	storagePool is the ScaleIO Storage Pool associated with the protection domain.
system	**string**	system is the name of the storage system as configured in ScaleIO.
volumeName	**string**	volumeName is the name of a volume already created in the ScaleIO system that is associated with this volume source.

5.1.41. .spec.scaleIO.secretRef

Description

SecretReference represents a Secret Reference. It has enough information to retrieve secret in any namespace

Type

object

Property	Type	Description
name	**string**	name is unique within a namespace to reference a secret resource.

Property	Type	Description
namespace	**string**	namespace defines the space within which the secret name must be unique.

5.1.42. .spec.storageos

Description

Represents a StorageOS persistent volume resource.

Type

object

Property	Type	Description
fsType	**string**	fsType is the filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
readOnly	**boolean**	readOnly defaults to false (read/write). ReadOnly here will force the ReadOnly setting in VolumeMounts.
secretRef	**object**	ObjectReference contains enough information to let you inspect or modify the referred object.
volumeName	**string**	volumeName is the human-readable name of the StorageOS volume. Volume names are only unique within a namespace.

Property	Type	Description
volumeNamespace	**string**	volumeNamespace specifies the scope of the volume within StorageOS. If no namespace is specified then the Pod's namespace will be used. This allows the Kubernetes name scoping to be mirrored within StorageOS for tighter integration. Set VolumeName to any name to override the default behaviour. Set to "default" if you are not using namespaces within StorageOS. Namespaces that do not pre-exist within StorageOS will be created.

5.1.43. .spec.storageos.secretRef

Description

ObjectReference contains enough information to let you inspect or modify the referred object.

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.

Property	Type	Description
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
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

5.1.44. .spec.vsphereVolume

Description

Represents a vSphere volume resource.

Type

object

Required

- **volumePath**

Property	Type	Description

Property	Type	Description
fsType	**string**	fsType is filesystem type to mount. Must be a filesystem type supported by the host operating system. Ex. "ext4", "xfs", "ntfs". Implicitly inferred to be "ext4" if unspecified.
storagePolicyID	**string**	storagePolicyID is the storage Policy Based Management (SPBM) profile ID associated with the StoragePolicyName.
storagePolicyName	**string**	storagePolicyName is the storage Policy Based Management (SPBM) profile name.
volumePath	**string**	volumePath is the path that identifies vSphere volume vmdk

5.1.45. .status

Description

PersistentVolumeStatus is the current status of a persistent volume.

Type

object

Property	Type	Description
lastPhaseTransitionTime	Time	lastPhaseTransitionTime is the time the phase transitioned from one to another and automatically resets to current time everytime a volume phase transitions.
message	**string**	message is a human-readable message indicating details about why the volume is in this state.

Property	Type	Description
phase	**string**	phase indicates if a volume is available, bound to a claim, or released by a claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#phase Possible enum values: - **"Available"** used for PersistentVolumes that are not yet bound Available volumes are held by the binder and matched to PersistentVolumeClaims - **"Bound"** used for PersistentVolumes that are bound - **"Failed"** used for PersistentVolumes that failed to be correctly recycled or deleted after being released from a claim - **"Pending"** used for PersistentVolumes that are not available - **"Released"** used for PersistentVolumes where the bound PersistentVolumeClaim was deleted released volumes must be recycled before becoming available again this phase is used by the persistent volume claim binder to signal to another process to reclaim the resource
reason	**string**	reason is a brief CamelCase string that describes any failure and is meant for machine parsing and tidy display in the CLI.

5.2. API ENDPOINTS

The following API endpoints are available:

- **/api/v1/persistentvolumes**
  - **DELETE**: delete collection of PersistentVolume
  - **GET**: list or watch objects of kind PersistentVolume
  - **POST**: create a PersistentVolume
- **/api/v1/watch/persistentvolumes**

  - **GET**: watch individual changes to a list of PersistentVolume. deprecated: use the 'watch' parameter with a list operation instead.

- **/api/v1/persistentvolumes/{name}**
  - **DELETE**: delete a PersistentVolume
  - **GET**: read the specified PersistentVolume
  - **PATCH**: partially update the specified PersistentVolume
  - **PUT**: replace the specified PersistentVolume
- **/api/v1/watch/persistentvolumes/{name}**
  - **GET**: watch changes to an object of kind PersistentVolume. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/api/v1/persistentvolumes/{name}/status**
  - **GET**: read status of the specified PersistentVolume
  - **PATCH**: partially update status of the specified PersistentVolume
  - **PUT**: replace status of the specified PersistentVolume

5.2.1. /api/v1/persistentvolumes

HTTP method

DELETE

Description

delete collection of PersistentVolume

Table 5.1. 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

Table 5.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind PersistentVolume

Table 5.3. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a PersistentVolume

Table 5.4. 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.

Table 5.5. Body parameters

Parameter	Type	Description
body	**PersistentVolume** schema	

Table 5.6. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
201 - Created	**PersistentVolume** schema
202 - Accepted	**PersistentVolume** schema
401 - Unauthorized	Empty

5.2.2. /api/v1/watch/persistentvolumes

HTTP method

GET

Description

watch individual changes to a list of PersistentVolume. deprecated: use the 'watch' parameter with a list operation instead.

Table 5.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

5.2.3. /api/v1/persistentvolumes/{name}

Table 5.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the PersistentVolume

HTTP method

DELETE

Description

delete a PersistentVolume

Table 5.9. 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

Table 5.10. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
202 - Accepted	**PersistentVolume** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified PersistentVolume

Table 5.11. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified PersistentVolume

Table 5.12. 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.

Table 5.13. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
201 - Created	**PersistentVolume** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified PersistentVolume

Table 5.14. 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.

Table 5.15. Body parameters

Parameter	Type	Description
body	**PersistentVolume** schema	

Table 5.16. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
201 - Created	**PersistentVolume** schema
401 - Unauthorized	Empty

5.2.4. /api/v1/watch/persistentvolumes/{name}

Table 5.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the PersistentVolume

HTTP method

GET

Description

watch changes to an object of kind PersistentVolume. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 5.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

5.2.5. /api/v1/persistentvolumes/{name}/status

Table 5.19. Global path parameters

Parameter	Type	Description
name	**string**	name of the PersistentVolume

HTTP method

GET

Description

read status of the specified PersistentVolume

Table 5.20. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified PersistentVolume

Table 5.21. Query parameters

Parameter	Type	Description

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.

Table 5.22. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
201 - Created	**PersistentVolume** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified PersistentVolume

Table 5.23. 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.

Table 5.24. Body parameters

Parameter	Type	Description
body	**PersistentVolume** schema	

Table 5.25. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolume** schema
201 - Created	**PersistentVolume** schema
401 - Unauthorized	Empty

CHAPTER 6. PERSISTENTVOLUMECLAIM [V1]

Description

PersistentVolumeClaim is a user's request for and claim to a persistent volume

Type

object

6.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	PersistentVolumeClaimSpec describes the common attributes of storage devices and allows a Source for provider-specific attributes
status	**object**	PersistentVolumeClaimStatus is the current status of a persistent volume claim.

6.1.1. .spec

Description

PersistentVolumeClaimSpec describes the common attributes of storage devices and allows a Source for provider-specific attributes

Type

object

Property	Type	Description
accessModes	**array (string)**	accessModes contains the desired access modes the volume should have. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1
dataSource	**object**	TypedLocalObjectReference contains enough information to let you locate the typed referenced object inside the same namespace.

Property	Type	Description
dataSourceRef	**object**	dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Property	Type	Description
resources	**object**	VolumeResourceRequirements describes the storage resource requirements for a volume.
selector	**LabelSelector**	selector is a label query over volumes to consider for binding.
storageClassName	**string**	storageClassName is the name of the StorageClass required by the claim. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#class-1

Property	Type	Description
volumeAttributesClassName	**string**	volumeAttributesClassName may be used to set the VolumeAttributesClass used by this claim. If specified, the CSI driver will create or update the volume with the attributes defined in the corresponding VolumeAttributesClass. This has a different purpose than storageClassName, it can be changed after the claim is created. An empty string value means that no VolumeAttributesClass will be applied to the claim but it's not allowed to reset this field to empty string once it is set. If unspecified and the PersistentVolumeClaim is unbound, the default VolumeAttributesClass will be set by the persistentvolume controller if it exists. If the resource referred to by volumeAttributesClass does not exist, this PersistentVolumeClaim will be set to a Pending state, as reflected by the modifyVolumeStatus field, until such as a resource exists. More info: https://kubernetes.io/docs/concepts/storage/volume-attributes-classes/ (Beta) Using this field requires the VolumeAttributesClass feature gate to be enabled (off by default).
volumeMode	**string**	volumeMode defines what type of volume is required by the claim. Value of Filesystem is implied when not included in claim spec. Possible enum values: - **"Block"** means the volume will not be formatted with a filesystem and will remain a raw block device. - **"Filesystem"** means the volume will be or is formatted with a filesystem.

Property	Type	Description
volumeName	**string**	volumeName is the binding reference to the PersistentVolume backing this claim.

6.1.2. .spec.dataSource

Description

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

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

6.1.3. .spec.dataSourceRef

Description

dataSourceRef specifies the object from which to populate the volume with data, if a non-empty volume is desired. This may be any object from a non-empty API group (non core object) or a PersistentVolumeClaim object. When this field is specified, volume binding will only succeed if the type of the specified object matches some installed volume populator or dynamic provisioner. This field will replace the functionality of the dataSource field and as such if both fields are non-empty, they must have the same value. For backwards compatibility, when namespace isn't specified in dataSourceRef, both fields (dataSource and dataSourceRef) will be set to the same value automatically if one of them is empty and the other is non-empty. When namespace is specified in dataSourceRef, dataSource isn't set to the same value and must be empty. There are three important differences between dataSource and dataSourceRef: * While dataSource only allows two specific types of objects, dataSourceRef allows any non-core object, as well as

PersistentVolumeClaim objects. * While dataSource ignores disallowed values (dropping them), dataSourceRef preserves all values, and generates an error if a disallowed value is specified. * While dataSource only allows local objects, dataSourceRef allows objects in any namespaces. (Beta) Using this field requires the AnyVolumeDataSource feature gate to be enabled. (Alpha) Using the namespace field of dataSourceRef requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

Type

object

Required

- **kind**
- **name**

Property	Type	Description
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
namespace	**string**	Namespace is the namespace of resource being referenced Note that when a namespace is specified, a gateway.networking.k8s.io/ReferenceGrant object is required in the referent namespace to allow that namespace's owner to accept the reference. See the ReferenceGrant documentation for details. (Alpha) This field requires the CrossNamespaceVolumeDataSource feature gate to be enabled.

6.1.4. .spec.resources

Description

VolumeResourceRequirements describes the storage resource requirements for a volume.

Type

object

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

6.1.5. .status

Description

PersistentVolumeClaimStatus is the current status of a persistent volume claim.

Type

object

Property	Type	Description
accessModes	**array (string)**	accessModes contains the actual access modes the volume backing the PVC has. More info: https://kubernetes.io/docs/concepts/storage/persistent-volumes#access-modes-1
allocatedResourceStatuses	**object (string)**	allocatedResourceStatuses stores status of resource being resized for the given PVC. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are

Property	Type	Description
		unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. ClaimResourceStatus can be in any of following states: - ControllerResizeInProgress: State set when resize controller starts resizing the volume in control-plane. - ControllerResizeFailed: State set when resize has failed in resize controller with a terminal error. - NodeResizePending: State set when resize controller has finished resizing the volume but further resizing of volume is needed on the node. - NodeResizeInProgress: State set when kubelet starts resizing the volume. - NodeResizeFailed: State set when resizing has failed in kubelet with a terminal error. Transient errors don't set NodeResizeFailed. For example: if expanding a PVC for more capacity - this field can be one of the following states: - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "ControllerResizeFailed" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizePending" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeInProgress" - pvc.status.allocatedResourceStatus['storage'] = "NodeResizeFailed" When this field is not set, it means that no resize operation is in progress for the given PVC. A controller that receives PVC update with previously unknown resourceName or ClaimResourceStatus should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC.

Property	Type	Description
		This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.
allocatedResources	**object (Quantity)**	allocatedResources tracks the resources allocated to a PVC including its capacity. Key names follow standard Kubernetes label syntax. Valid values are either: * Un-prefixed keys: - storage - the capacity of the volume. * Custom resources must use implementation-defined prefixed names such as "example.com/my-custom-resource" Apart from above values - keys that are unprefixed or have kubernetes.io prefix are considered reserved and hence may not be used. Capacity reported here may be larger than the actual capacity when a volume expansion operation is requested. For storage quota, the larger value from allocatedResources and PVC.spec.resources is used. If allocatedResources is not set, PVC.spec.resources alone is used for quota calculation. If a volume expansion capacity request is lowered, allocatedResources is only lowered if there are no expansion operations in progress and if the actual volume capacity is equal or lower than the requested capacity. A controller that receives PVC update with previously unknown resourceName should ignore the update for the purpose it was designed. For example - a controller that only is responsible for resizing capacity of the volume, should ignore PVC updates that change other valid resources associated with PVC. This is an alpha field and requires enabling RecoverVolumeExpansionFailure feature.

Property	Type	Description
capacity	**object (Quantity)**	capacity represents the actual resources of the underlying volume.
conditions	**array**	conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.
conditions[]	**object**	PersistentVolumeClaimCondition contains details about state of pvc
currentVolumeAttributesClassName	**string**	currentVolumeAttributesClassName is the current name of the VolumeAttributesClass the PVC is using. When unset, there is no VolumeAttributeClass applied to this PersistentVolumeClaim This is a beta field and requires enabling VolumeAttributesClass feature (off by default).
modifyVolumeStatus	**object**	ModifyVolumeStatus represents the status object of ControllerModifyVolume operation
phase	**string**	phase represents the current phase of PersistentVolumeClaim. Possible enum values: - **"Bound"** used for PersistentVolumeClaims that are bound - **"Lost"** used for PersistentVolumeClaims that lost their underlying PersistentVolume. The claim was bound to a PersistentVolume and this volume does not exist any longer and all data on it was lost. - **"Pending"** used for PersistentVolumeClaims that are not yet bound

6.1.6. .status.conditions

Description

conditions is the current Condition of persistent volume claim. If underlying persistent volume is being resized then the Condition will be set to 'Resizing'.

Type

array

6.1.7. .status.conditions[]

Description

PersistentVolumeClaimCondition contains details about state of pvc

Type

object

Required

- **type**
- **status**

Property	Type	Description
lastProbeTime	Time	lastProbeTime is the time we probed the condition.
lastTransitionTime	Time	lastTransitionTime is the time the condition transitioned from one status to another.
message	**string**	message is the human-readable message indicating details about last transition.
reason	**string**	reason is a unique, this should be a short, machine understandable string that gives the reason for condition's last transition. If it reports "Resizing" that means the underlying persistent volume is being resized.
status	**string**	
type	**string**	

6.1.8. .status.modifyVolumeStatus

Description

ModifyVolumeStatus represents the status object of ControllerModifyVolume operation

Type

object

Required

- **status**

Property	Type	Description
status	**string**	status is the status of the ControllerModifyVolume operation. It can be in any of following states: - Pending Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing. - InProgress InProgress indicates that the volume is being modified. - Infeasible Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified. Note: New statuses can be added in the future. Consumers should check for unknown statuses and fail appropriately. Possible enum values: - **"InProgress"** InProgress indicates that the volume is being modified - **"Infeasible"** Infeasible indicates that the request has been rejected as invalid by the CSI driver. To resolve the error, a valid VolumeAttributesClass needs to be specified - **"Pending"** Pending indicates that the PersistentVolumeClaim cannot be modified due to unmet requirements, such as the specified VolumeAttributesClass not existing
targetVolumeAttributesClassName	**string**	targetVolumeAttributesClassName is the name of the VolumeAttributesClass the PVC currently being reconciled

6.2. API ENDPOINTS

The following API endpoints are available:

- **/api/v1/persistentvolumeclaims**
  - **GET**: list or watch objects of kind PersistentVolumeClaim
- **/api/v1/watch/persistentvolumeclaims**
  - **GET**: watch individual changes to a list of PersistentVolumeClaim. deprecated: use the 'watch' parameter with a list operation instead.
- **/api/v1/namespaces/{namespace}/persistentvolumeclaims**
  - **DELETE**: delete collection of PersistentVolumeClaim
  - **GET**: list or watch objects of kind PersistentVolumeClaim
  - **POST**: create a PersistentVolumeClaim
- **/api/v1/watch/namespaces/{namespace}/persistentvolumeclaims**
  - **GET**: watch individual changes to a list of PersistentVolumeClaim. deprecated: use the 'watch' parameter with a list operation instead.
- **/api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}**
  - **DELETE**: delete a PersistentVolumeClaim
  - **GET**: read the specified PersistentVolumeClaim
  - **PATCH**: partially update the specified PersistentVolumeClaim
  - **PUT**: replace the specified PersistentVolumeClaim
- **/api/v1/watch/namespaces/{namespace}/persistentvolumeclaims/{name}**
  - **GET**: watch changes to an object of kind PersistentVolumeClaim. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}/status**
  - **GET**: read status of the specified PersistentVolumeClaim
  - **PATCH**: partially update status of the specified PersistentVolumeClaim
  - **PUT**: replace status of the specified PersistentVolumeClaim

6.2.1. /api/v1/persistentvolumeclaims

HTTP method

GET

Description

list or watch objects of kind PersistentVolumeClaim

Table 6.1. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaimList** schema
401 - Unauthorized	Empty

6.2.2. /api/v1/watch/persistentvolumeclaims

HTTP method

GET

Description

watch individual changes to a list of PersistentVolumeClaim. deprecated: use the 'watch' parameter with a list operation instead.

Table 6.2. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

6.2.3. /api/v1/namespaces/{namespace}/persistentvolumeclaims

HTTP method

DELETE

Description

delete collection of PersistentVolumeClaim

Table 6.3. 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

Table 6.4. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind PersistentVolumeClaim

Table 6.5. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaimList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a PersistentVolumeClaim

Table 6.6. 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.

Table 6.7. Body parameters

Parameter	Type	Description
body	**PersistentVolumeClaim** schema	

Table 6.8. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
201 - Created	**PersistentVolumeClaim** schema
202 - Accepted	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

6.2.4. /api/v1/watch/namespaces/{namespace}/persistentvolumeclaims

HTTP method

GET

Description

watch individual changes to a list of PersistentVolumeClaim. deprecated: use the 'watch' parameter with a list operation instead.

Table 6.9. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

6.2.5. /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}

Table 6.10. Global path parameters

Parameter	Type	Description
name	**string**	name of the PersistentVolumeClaim

HTTP method

DELETE

Description

delete a PersistentVolumeClaim

Table 6.11. 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

Table 6.12. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
202 - Accepted	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified PersistentVolumeClaim

Table 6.13. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified PersistentVolumeClaim

Table 6.14. 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.

Table 6.15. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
201 - Created	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified PersistentVolumeClaim

Table 6.16. 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.

Table 6.17. Body parameters

Parameter	Type	Description
body	**PersistentVolumeClaim** schema	

Table 6.18. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
201 - Created	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

6.2.6. /api/v1/watch/namespaces/{namespace}/persistentvolumeclaims/{name}

Table 6.19. Global path parameters

Parameter	Type	Description
name	**string**	name of the PersistentVolumeClaim

HTTP method

GET

Description

watch changes to an object of kind PersistentVolumeClaim. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 6.20. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

6.2.7. /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}/status

Table 6.21. Global path parameters

Parameter	Type	Description
name	**string**	name of the PersistentVolumeClaim

HTTP method

GET

Description

read status of the specified PersistentVolumeClaim

Table 6.22. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified PersistentVolumeClaim

Table 6.23. Query parameters

Parameter	Type	Description

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.

Table 6.24. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
201 - Created	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified PersistentVolumeClaim

Table 6.25. 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.

Table 6.26. Body parameters

Parameter	Type	Description
body	**PersistentVolumeClaim** schema	

Table 6.27. HTTP responses

HTTP code	Reponse body
200 - OK	**PersistentVolumeClaim** schema
201 - Created	**PersistentVolumeClaim** schema
401 - Unauthorized	Empty

CHAPTER 7. STORAGECLASS [STORAGE.K8S.IO/V1]

Description

StorageClass describes the parameters for a class of storage for which PersistentVolumes can be dynamically provisioned.
StorageClasses are non-namespaced; the name of the storage class according to etcd is in ObjectMeta.Name.

Type

object

Required

- **provisioner**

7.1. SPECIFICATION

Property	Type	Description
allowVolumeExpansion	**boolean**	allowVolumeExpansion shows whether the storage class allow volume expand.
allowedTopologies	**array (TopologySelectorTerm)**	allowedTopologies restrict the node topologies where volumes can be dynamically provisioned. Each volume plugin defines its own supported topology specifications. An empty TopologySelectorTerm list means there is no topology restriction. This field is only honored by servers that enable the VolumeScheduling feature.
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**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
mountOptions	**array (string)**	mountOptions controls the mountOptions for dynamically provisioned PersistentVolumes of this storage class. e.g. ["ro", "soft"]. Not validated - mount of the PVs will simply fail if one is invalid.
parameters	**object (string)**	parameters holds the parameters for the provisioner that should create volumes of this storage class.
provisioner	**string**	provisioner indicates the type of the provisioner.

Property	Type	Description
reclaimPolicy	**string**	reclaimPolicy controls the reclaimPolicy for dynamically provisioned PersistentVolumes of this storage class. Defaults to Delete. Possible enum values: - **"Delete"** means the volume will be deleted from Kubernetes on release from its claim. The volume plugin must support Deletion. - **"Recycle"** means the volume will be recycled back into the pool of unbound persistent volumes on release from its claim. The volume plugin must support Recycling. - **"Retain"** means the volume will be left in its current phase (Released) for manual reclamation by the administrator. The default policy is Retain.
volumeBindingMode	**string**	volumeBindingMode indicates how PersistentVolumeClaims should be provisioned and bound. When unset, VolumeBindingImmediate is used. This field is only honored by servers that enable the VolumeScheduling feature. Possible enum values: - **"Immediate"** indicates that PersistentVolumeClaims should be immediately provisioned and bound. This is the default mode. - **"WaitForFirstConsumer"** indicates that PersistentVolumeClaims should not be provisioned and bound until the first Pod is created that references the PeristentVolumeClaim. The volume provisioning and binding will occur during Pod scheduing.

7.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/storage.k8s.io/v1/storageclasses**

  - **DELETE**: delete collection of StorageClass
  - **GET**: list or watch objects of kind StorageClass
  - **POST**: create a StorageClass
- **/apis/storage.k8s.io/v1/watch/storageclasses**
  - **GET**: watch individual changes to a list of StorageClass. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/storage.k8s.io/v1/storageclasses/{name}**
  - **DELETE**: delete a StorageClass
  - **GET**: read the specified StorageClass
  - **PATCH**: partially update the specified StorageClass
  - **PUT**: replace the specified StorageClass
- **/apis/storage.k8s.io/v1/watch/storageclasses/{name}**
  - **GET**: watch changes to an object of kind StorageClass. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

7.2.1. /apis/storage.k8s.io/v1/storageclasses

HTTP method

DELETE

Description

delete collection of StorageClass

Table 7.1. 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

Table 7.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind StorageClass

Table 7.3. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageClassList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a StorageClass

Table 7.4. 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.

Table 7.5. Body parameters

Parameter	Type	Description
body	**StorageClass** schema	

Table 7.6. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageClass** schema
201 - Created	**StorageClass** schema
202 - Accepted	**StorageClass** schema
401 - Unauthorized	Empty

7.2.2. /apis/storage.k8s.io/v1/watch/storageclasses

HTTP method

GET

Description

watch individual changes to a list of StorageClass. deprecated: use the 'watch' parameter with a list operation instead.

Table 7.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

7.2.3. /apis/storage.k8s.io/v1/storageclasses/{name}

Table 7.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the StorageClass

HTTP method

DELETE

Description

delete a StorageClass

Table 7.9. 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

Table 7.10. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageClass** schema
202 - Accepted	**StorageClass** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified StorageClass

Table 7.11. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageClass** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified StorageClass

Table 7.12. 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.

Table 7.13. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageClass** schema
201 - Created	**StorageClass** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified StorageClass

Table 7.14. 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.

Table 7.15. Body parameters

Parameter	Type	Description
body	**StorageClass** schema	

Table 7.16. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageClass** schema
201 - Created	**StorageClass** schema
401 - Unauthorized	Empty

7.2.4. /apis/storage.k8s.io/v1/watch/storageclasses/{name}

Table 7.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the StorageClass

HTTP method

GET

Description

watch changes to an object of kind StorageClass. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 7.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

CHAPTER 8. STORAGESTATE [MIGRATION.K8S.IO/V1ALPHA1]

Description

The state of the storage of a specific resource.

Type

object

8.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	Specification of the storage state.
status	**object**	Status of the storage state.

8.1.1. .spec

Description

Specification of the storage state.

Type

object

Property	Type	Description
resource	**object**	The resource this storageState is about.

8.1.2. .spec.resource

Description

The resource this storageState is about.

Type

object

Property	Type	Description
group	**string**	The name of the group.
resource	**string**	The name of the resource.

8.1.3. .status

Description

Status of the storage state.

Type

object

Property	Type	Description
currentStorageVersionHash	**string**	The hash value of the current storage version, as shown in the discovery document served by the API server. Storage Version is the version to which objects are converted to before persisted.
lastHeartbeatTime	**string**	LastHeartbeatTime is the last time the storage migration triggering controller checks the storage version hash of this resource in the discovery document and updates this field.

Property	Type	Description
persistedStorageVersionHashes	**array (string)**	The hash values of storage versions that persisted instances of spec.resource might still be encoded in. "Unknown" is a valid value in the list, and is the default value. It is not safe to upgrade or downgrade to an apiserver binary that does not support all versions listed in this field, or if "Unknown" is listed. Once the storage version migration for this resource has completed, the value of this field is refined to only contain the currentStorageVersionHash. Once the apiserver has changed the storage version, the new storage version is appended to the list.

8.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/migration.k8s.io/v1alpha1/storagestates**
  - **DELETE**: delete collection of StorageState
  - **GET**: list objects of kind StorageState
  - **POST**: create a StorageState
- **/apis/migration.k8s.io/v1alpha1/storagestates/{name}**
  - **DELETE**: delete a StorageState
  - **GET**: read the specified StorageState
  - **PATCH**: partially update the specified StorageState
  - **PUT**: replace the specified StorageState
- **/apis/migration.k8s.io/v1alpha1/storagestates/{name}/status**
  - **GET**: read status of the specified StorageState
  - **PATCH**: partially update status of the specified StorageState
  - **PUT**: replace status of the specified StorageState

8.2.1. /apis/migration.k8s.io/v1alpha1/storagestates

HTTP method

DELETE

Description

delete collection of StorageState

Table 8.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind StorageState

Table 8.2. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageStateList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a StorageState

Table 8.3. 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.

Table 8.4. Body parameters

Parameter	Type	Description
body	**StorageState** schema	

Table 8.5. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
201 - Created	**StorageState** schema
202 - Accepted	**StorageState** schema
401 - Unauthorized	Empty

8.2.2. /apis/migration.k8s.io/v1alpha1/storagestates/{name}

Table 8.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the StorageState

HTTP method

DELETE

Description

delete a StorageState

Table 8.7. 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

Table 8.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified StorageState

Table 8.9. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified StorageState

Table 8.10. 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.

Table 8.11. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified StorageState

Table 8.12. 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.

Table 8.13. Body parameters

Parameter	Type	Description
body	**StorageState** schema	

Table 8.14. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
201 - Created	**StorageState** schema
401 - Unauthorized	Empty

8.2.3. /apis/migration.k8s.io/v1alpha1/storagestates/{name}/status

Table 8.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the StorageState

HTTP method

GET

Description

read status of the specified StorageState

Table 8.16. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified StorageState

Table 8.17. 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.

Table 8.18. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified StorageState

Table 8.19. 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.

Table 8.20. Body parameters

Parameter	Type	Description
body	**StorageState** schema	

Table 8.21. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageState** schema
201 - Created	**StorageState** schema
401 - Unauthorized	Empty

CHAPTER 9. STORAGEVERSIONMIGRATION [MIGRATION.K8S.IO/V1ALPHA1]

Description

StorageVersionMigration represents a migration of stored data to the latest storage version.

Type

object

9.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	Specification of the migration.
status	**object**	Status of the migration.

9.1.1. .spec

Description

Specification of the migration.

Type

object

Required

- **resource**

Property	Type	Description
continueToken	**string**	The token used in the list options to get the next chunk of objects to migrate. When the .status.conditions indicates the migration is "Running", users can use this token to check the progress of the migration.
resource	**object**	The resource that is being migrated. The migrator sends requests to the endpoint serving the resource. Immutable.

9.1.2. .spec.resource

Description

The resource that is being migrated. The migrator sends requests to the endpoint serving the resource. Immutable.

Type

object

Property	Type	Description
group	**string**	The name of the group.
resource	**string**	The name of the resource.
version	**string**	The name of the version.

9.1.3. .status

Description

Status of the migration.

Type

object

Property	Type	Description
conditions	**array**	The latest available observations of the migration's current state.
conditions[]	**object**	Describes the state of a migration at a certain point.

9.1.4. .status.conditions

Description

The latest available observations of the migration's current state.

Type

array

9.1.5. .status.conditions[]

Description

Describes the state of a migration at a certain point.

Type

object

Required

- **status**
- **type**

Property	Type	Description
lastUpdateTime	**string**	The last time this condition was updated.
message	**string**	A human readable message indicating details about the transition.
reason	**string**	The reason for the condition's last transition.
status	**string**	Status of the condition, one of True, False, Unknown.
type	**string**	Type of the condition.

9.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/migration.k8s.io/v1alpha1/storageversionmigrations**
  - **DELETE**: delete collection of StorageVersionMigration
  - **GET**: list objects of kind StorageVersionMigration
  - **POST**: create a StorageVersionMigration
- **/apis/migration.k8s.io/v1alpha1/storageversionmigrations/{name}**
  - **DELETE**: delete a StorageVersionMigration
  - **GET**: read the specified StorageVersionMigration
  - **PATCH**: partially update the specified StorageVersionMigration
  - **PUT**: replace the specified StorageVersionMigration
- **/apis/migration.k8s.io/v1alpha1/storageversionmigrations/{name}/status**
  - **GET**: read status of the specified StorageVersionMigration
  - **PATCH**: partially update status of the specified StorageVersionMigration
  - **PUT**: replace status of the specified StorageVersionMigration

9.2.1. /apis/migration.k8s.io/v1alpha1/storageversionmigrations

HTTP method

DELETE

Description

delete collection of StorageVersionMigration

Table 9.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind StorageVersionMigration

Table 9.2. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigrationList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a StorageVersionMigration

Table 9.3. 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.

Table 9.4. Body parameters

Parameter	Type	Description
body	**StorageVersionMigration** schema	

Table 9.5. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
201 - Created	**StorageVersionMigration** schema
202 - Accepted	**StorageVersionMigration** schema
401 - Unauthorized	Empty

9.2.2. /apis/migration.k8s.io/v1alpha1/storageversionmigrations/{name}

Table 9.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the StorageVersionMigration

HTTP method

DELETE

Description

delete a StorageVersionMigration

Table 9.7. 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

Table 9.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified StorageVersionMigration

Table 9.9. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified StorageVersionMigration

Table 9.10. 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.

Table 9.11. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified StorageVersionMigration

Table 9.12. 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.

Table 9.13. Body parameters

Parameter	Type	Description
body	**StorageVersionMigration** schema	

Table 9.14. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
201 - Created	**StorageVersionMigration** schema
401 - Unauthorized	Empty

9.2.3. /apis/migration.k8s.io/v1alpha1/storageversionmigrations/{name}/status

Table 9.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the StorageVersionMigration

HTTP method

GET

Description

read status of the specified StorageVersionMigration

Table 9.16. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified StorageVersionMigration

Table 9.17. 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.

Table 9.18. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified StorageVersionMigration

Table 9.19. 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.

Table 9.20. Body parameters

Parameter	Type	Description
body	**StorageVersionMigration** schema	

Table 9.21. HTTP responses

HTTP code	Reponse body
200 - OK	**StorageVersionMigration** schema
201 - Created	**StorageVersionMigration** schema
401 - Unauthorized	Empty

CHAPTER 10. VOLUMEATTACHMENT [STORAGE.K8S.IO/V1]

Description

VolumeAttachment captures the intent to attach or detach the specified volume to/from the specified node.
VolumeAttachment objects are non-namespaced.

Type

object

Required

- **spec**

10.1. 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
metadata	**ObjectMeta**	Standard object metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	VolumeAttachmentSpec is the specification of a VolumeAttachment request.

Property	Type	Description
status	**object**	VolumeAttachmentStatus is the status of a VolumeAttachment request.

10.1.1. .spec

Description

VolumeAttachmentSpec is the specification of a VolumeAttachment request.

Type

object

Required

- **attacher**
- **source**
- **nodeName**

Property	Type	Description
attacher	**string**	attacher indicates the name of the volume driver that MUST handle this request. This is the name returned by GetPluginName().
nodeName	**string**	nodeName represents the node that the volume should be attached to.
source	**object**	VolumeAttachmentSource represents a volume that should be attached. Right now only PersistenVolumes can be attached via external attacher, in future we may allow also inline volumes in pods. Exactly one member can be set.

10.1.2. .spec.source

Description

VolumeAttachmentSource represents a volume that should be attached. Right now only PersistenVolumes can be attached via external attacher, in future we may allow also inline volumes in pods. Exactly one member can be set.

Type

object

Property	Type	Description
inlineVolumeSpec	**PersistentVolumeSpec**	inlineVolumeSpec contains all the information necessary to attach a persistent volume defined by a pod's inline VolumeSource. This field is populated only for the CSIMigration feature. It contains translated fields from a pod's inline VolumeSource to a PersistentVolumeSpec. This field is beta-level and is only honored by servers that enabled the CSIMigration feature.
persistentVolumeName	**string**	persistentVolumeName represents the name of the persistent volume to attach.

10.1.3. .status

Description

VolumeAttachmentStatus is the status of a VolumeAttachment request.

Type

object

Required

- **attached**

Property	Type	Description
attachError	**object**	VolumeError captures an error encountered during a volume operation.
attached	**boolean**	attached indicates the volume is successfully attached. This field must only be set by the entity completing the attach operation, i.e. the external-attacher.

Property	Type	Description
attachmentMetadata	**object (string)**	attachmentMetadata is populated with any information returned by the attach operation, upon successful attach, that must be passed into subsequent WaitForAttach or Mount calls. This field must only be set by the entity completing the attach operation, i.e. the external-attacher.
detachError	**object**	VolumeError captures an error encountered during a volume operation.

10.1.4. .status.attachError

Description

VolumeError captures an error encountered during a volume operation.

Type

object

Property	Type	Description
message	**string**	message represents the error encountered during Attach or Detach operation. This string may be logged, so it should not contain sensitive information.
time	**Time**	time represents the time the error was encountered.

10.1.5. .status.detachError

Description

VolumeError captures an error encountered during a volume operation.

Type

object

Property	Type	Description

Property	Type	Description
message	**string**	message represents the error encountered during Attach or Detach operation. This string may be logged, so it should not contain sensitive information.
time	**Time**	time represents the time the error was encountered.

10.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/storage.k8s.io/v1/volumeattachments**
  - **DELETE**: delete collection of VolumeAttachment
  - **GET**: list or watch objects of kind VolumeAttachment
  - **POST**: create a VolumeAttachment
- **/apis/storage.k8s.io/v1/watch/volumeattachments**
  - **GET**: watch individual changes to a list of VolumeAttachment. deprecated: use the 'watch' parameter with a list operation instead.
- **/apis/storage.k8s.io/v1/volumeattachments/{name}**
  - **DELETE**: delete a VolumeAttachment
  - **GET**: read the specified VolumeAttachment
  - **PATCH**: partially update the specified VolumeAttachment
  - **PUT**: replace the specified VolumeAttachment
- **/apis/storage.k8s.io/v1/watch/volumeattachments/{name}**
  - **GET**: watch changes to an object of kind VolumeAttachment. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.
- **/apis/storage.k8s.io/v1/volumeattachments/{name}/status**
  - **GET**: read status of the specified VolumeAttachment
  - **PATCH**: partially update status of the specified VolumeAttachment
  - **PUT**: replace status of the specified VolumeAttachment

10.2.1. /apis/storage.k8s.io/v1/volumeattachments

HTTP method

DELETE

Description

delete collection of VolumeAttachment

Table 10.1. 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

Table 10.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list or watch objects of kind VolumeAttachment

Table 10.3. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachmentList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a VolumeAttachment

Table 10.4. 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.

Table 10.5. Body parameters

Parameter	Type	Description
body	**VolumeAttachment** schema	

Table 10.6. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
201 - Created	**VolumeAttachment** schema
202 - Accepted	**VolumeAttachment** schema
401 - Unauthorized	Empty

10.2.2. /apis/storage.k8s.io/v1/watch/volumeattachments

HTTP method

GET

Description

watch individual changes to a list of VolumeAttachment. deprecated: use the 'watch' parameter with a list operation instead.

Table 10.7. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

10.2.3. /apis/storage.k8s.io/v1/volumeattachments/{name}

Table 10.8. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeAttachment

HTTP method

DELETE

Description

delete a VolumeAttachment

Table 10.9. 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

Table 10.10. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
202 - Accepted	**VolumeAttachment** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified VolumeAttachment

Table 10.11. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified VolumeAttachment

Table 10.12. 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.

Table 10.13. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
201 - Created	**VolumeAttachment** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified VolumeAttachment

Table 10.14. 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.

Table 10.15. Body parameters

Parameter	Type	Description
body	**VolumeAttachment** schema	

Table 10.16. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
201 - Created	**VolumeAttachment** schema
401 - Unauthorized	Empty

10.2.4. /apis/storage.k8s.io/v1/watch/volumeattachments/{name}

Table 10.17. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeAttachment

HTTP method

GET

Description

watch changes to an object of kind VolumeAttachment. deprecated: use the 'watch' parameter with a list operation instead, filtered to a single item with the 'fieldSelector' parameter.

Table 10.18. HTTP responses

HTTP code	Reponse body
200 - OK	**WatchEvent** schema
401 - Unauthorized	Empty

10.2.5. /apis/storage.k8s.io/v1/volumeattachments/{name}/status

Table 10.19. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeAttachment

HTTP method

GET

Description

read status of the specified VolumeAttachment

Table 10.20. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified VolumeAttachment

Table 10.21. Query parameters

Parameter	Type	Description

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.

Table 10.22. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
201 - Created	**VolumeAttachment** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified VolumeAttachment

Table 10.23. Query parameters

Parameter	Type	Description

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.

Table 10.24. Body parameters

Parameter	Type	Description
body	**VolumeAttachment** schema	

Table 10.25. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeAttachment** schema
201 - Created	**VolumeAttachment** schema
401 - Unauthorized	Empty

CHAPTER 11. VOLUMESNAPSHOT [SNAPSHOT.STORAGE.K8S.IO/V1]

Description

VolumeSnapshot is a user's request for either creating a point-in-time snapshot of a persistent volume, or binding to a pre-existing snapshot.

Type

object

Required

- **spec**

11.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

Property	Type	Description
spec	**object**	spec defines the desired characteristics of a snapshot requested by a user. More info: https://kubernetes.io/docs/concepts/storage/volume-snapshots#volumesnapshots Required.
status	**object**	status represents the current information of a snapshot. Consumers must verify binding between VolumeSnapshot and VolumeSnapshotContent objects is successful (by validating that both VolumeSnapshot and VolumeSnapshotContent point at each other) before using this object.

11.1.1. .spec

Description

spec defines the desired characteristics of a snapshot requested by a user. More info: https://kubernetes.io/docs/concepts/storage/volume-snapshots#volumesnapshots Required.

Type

object

Required

- **source**

Property	Type	Description
source	**object**	source specifies where a snapshot will be created from. This field is immutable after creation. Required.

Property	Type	Description
volumeSnapshotClassName	**string**	VolumeSnapshotClassName is the name of the VolumeSnapshotClass requested by the VolumeSnapshot. VolumeSnapshotClassName may be left nil to indicate that the default SnapshotClass should be used. A given cluster may have multiple default Volume SnapshotClasses: one default per CSI Driver. If a VolumeSnapshot does not specify a SnapshotClass, VolumeSnapshotSource will be checked to figure out what the associated CSI Driver is, and the default VolumeSnapshotClass associated with that CSI Driver will be used. If more than one VolumeSnapshotClass exist for a given CSI Driver and more than one have been marked as default, CreateSnapshot will fail and generate an event. Empty string is not allowed for this field.

11.1.2. .spec.source

Description

source specifies where a snapshot will be created from. This field is immutable after creation. Required.

Type

object

Property	Type	Description
persistentVolumeClaimName	**string**	persistentVolumeClaimName specifies the name of the PersistentVolumeClaim object representing the volume from which a snapshot should be created. This PVC is assumed to be in the same namespace as the VolumeSnapshot object. This field should be set if the snapshot does not exists, and needs to be created. This field is immutable.

Property	Type	Description
volumeSnapshotContentName	**string**	volumeSnapshotContentName specifies the name of a pre-existing VolumeSnapshotContent object representing an existing volume snapshot. This field should be set if the snapshot already exists and only needs a representation in Kubernetes. This field is immutable.

11.1.3. .status

Description

status represents the current information of a snapshot. Consumers must verify binding between VolumeSnapshot and VolumeSnapshotContent objects is successful (by validating that both VolumeSnapshot and VolumeSnapshotContent point at each other) before using this object.

Type

object

Property	Type	Description
boundVolumeSnapshotContentName	**string**	boundVolumeSnapshotContentName is the name of the VolumeSnapshotContent object to which this VolumeSnapshot object intends to bind to. If not specified, it indicates that the VolumeSnapshot object has not been successfully bound to a VolumeSnapshotContent object yet. NOTE: To avoid possible security issues, consumers must verify binding between VolumeSnapshot and VolumeSnapshotContent objects is successful (by validating that both VolumeSnapshot and VolumeSnapshotContent point at each other) before using this object.

Property	Type	Description
creationTime	**string**	creationTime is the timestamp when the point-in-time snapshot is taken by the underlying storage system. In dynamic snapshot creation case, this field will be filled in by the snapshot controller with the "creation_time" value returned from CSI "CreateSnapshot" gRPC call. For a pre-existing snapshot, this field will be filled with the "creation_time" value returned from the CSI "ListSnapshots" gRPC call if the driver supports it. If not specified, it may indicate that the creation time of the snapshot is unknown.
error	**object**	error is the last observed error during snapshot creation, if any. This field could be helpful to upper level controllers(i.e., application controller) to decide whether they should continue on waiting for the snapshot to be created based on the type of error reported. The snapshot controller will keep retrying when an error occurs during the snapshot creation. Upon success, this error field will be cleared.
readyToUse	**boolean**	readyToUse indicates if the snapshot is ready to be used to restore a volume. In dynamic snapshot creation case, this field will be filled in by the snapshot controller with the "ready_to_use" value returned from CSI "CreateSnapshot" gRPC call. For a pre-existing snapshot, this field will be filled with the "ready_to_use" value returned from the CSI "ListSnapshots" gRPC call if the driver supports it, otherwise, this field will be set to "True". If not specified, it means the readiness of a snapshot is unknown.

Property	Type	Description
restoreSize	**integer-or-string**	restoreSize represents the minimum size of volume required to create a volume from this snapshot. In dynamic snapshot creation case, this field will be filled in by the snapshot controller with the "size_bytes" value returned from CSI "CreateSnapshot" gRPC call. For a pre-existing snapshot, this field will be filled with the "size_bytes" value returned from the CSI "ListSnapshots" gRPC call if the driver supports it. When restoring a volume from this snapshot, the size of the volume MUST NOT be smaller than the restoreSize if it is specified, otherwise the restoration will fail. If not specified, it indicates that the size is unknown.
volumeGroupSnapshotName	**string**	VolumeGroupSnapshotName is the name of the VolumeGroupSnapshot of which this VolumeSnapshot is a part of.

11.1.4. .status.error

Description

error is the last observed error during snapshot creation, if any. This field could be helpful to upper level controllers(i.e., application controller) to decide whether they should continue on waiting for the snapshot to be created based on the type of error reported. The snapshot controller will keep retrying when an error occurs during the snapshot creation. Upon success, this error field will be cleared.

Type

object

Property	Type	Description
message	**string**	message is a string detailing the encountered error during snapshot creation if specified. NOTE: message may be logged, and it should not contain sensitive information.

Property	Type	Description
time	**string**	time is the timestamp when the error was encountered.

11.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/snapshot.storage.k8s.io/v1/volumesnapshots**
  - **GET**: list objects of kind VolumeSnapshot
- **/apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots**
  - **DELETE**: delete collection of VolumeSnapshot
  - **GET**: list objects of kind VolumeSnapshot
  - **POST**: create a VolumeSnapshot
- **/apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots/{name}**
  - **DELETE**: delete a VolumeSnapshot
  - **GET**: read the specified VolumeSnapshot
  - **PATCH**: partially update the specified VolumeSnapshot
  - **PUT**: replace the specified VolumeSnapshot
- **/apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots/{name}/status**
  - **GET**: read status of the specified VolumeSnapshot
  - **PATCH**: partially update status of the specified VolumeSnapshot
  - **PUT**: replace status of the specified VolumeSnapshot

11.2.1. /apis/snapshot.storage.k8s.io/v1/volumesnapshots

HTTP method

GET

Description

list objects of kind VolumeSnapshot

Table 11.1. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotList** schema

HTTP code	Reponse body
401 - Unauthorized	Empty

11.2.2. /apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots

HTTP method

DELETE

Description

delete collection of VolumeSnapshot

Table 11.2. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind VolumeSnapshot

Table 11.3. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a VolumeSnapshot

Table 11.4. 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.

Table 11.5. Body parameters

Parameter	Type	Description
body	**VolumeSnapshot** schema	

Table 11.6. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
201 - Created	**VolumeSnapshot** schema
202 - Accepted	**VolumeSnapshot** schema
401 - Unauthorized	Empty

11.2.3. /apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots/

Table 11.7. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeSnapshot

HTTP method

DELETE

Description

delete a VolumeSnapshot

Table 11.8. 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

Table 11.9. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified VolumeSnapshot

Table 11.10. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified VolumeSnapshot

Table 11.11. 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.

Table 11.12. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified VolumeSnapshot

Table 11.13. 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.

Table 11.14. Body parameters

Parameter	Type	Description
body	**VolumeSnapshot** schema	

Table 11.15. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
201 - Created	**VolumeSnapshot** schema
401 - Unauthorized	Empty

11.2.4. /apis/snapshot.storage.k8s.io/v1/namespaces/{namespace}/volumesnapshots/

Table 11.16. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeSnapshot

HTTP method

GET

Description

read status of the specified VolumeSnapshot

Table 11.17. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified VolumeSnapshot

Table 11.18. 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.

Table 11.19. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified VolumeSnapshot

Table 11.20. 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.

Table 11.21. Body parameters

Parameter	Type	Description
body	**VolumeSnapshot** schema	

Table 11.22. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshot** schema
201 - Created	**VolumeSnapshot** schema
401 - Unauthorized	Empty

CHAPTER 12. VOLUMESNAPSHOTCLASS [SNAPSHOT.STORAGE.K8S.IO/V1]

Description

VolumeSnapshotClass specifies parameters that a underlying storage system uses when creating a volume snapshot. A specific VolumeSnapshotClass is used by specifying its name in a VolumeSnapshot object. VolumeSnapshotClasses are non-namespaced

Type

object

Required

- **deletionPolicy**
- **driver**

12.1. 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
deletionPolicy	**string**	deletionPolicy determines whether a VolumeSnapshotContent created through the VolumeSnapshotClass should be deleted when its bound VolumeSnapshot is deleted. Supported values are "Retain" and "Delete". "Retain" means that the VolumeSnapshotContent and its physical snapshot on underlying storage system are kept. "Delete" means that the VolumeSnapshotContent and its physical snapshot on underlying storage system are deleted. Required.

Property	Type	Description
driver	**string**	driver is the name of the storage driver that handles this VolumeSnapshotClass. Required.
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**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
parameters	**object (string)**	parameters is a key-value map with storage driver specific parameters for creating snapshots. These values are opaque to Kubernetes.

12.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/snapshot.storage.k8s.io/v1/volumesnapshotclasses**
  - **DELETE**: delete collection of VolumeSnapshotClass
  - **GET**: list objects of kind VolumeSnapshotClass
  - **POST**: create a VolumeSnapshotClass
- **/apis/snapshot.storage.k8s.io/v1/volumesnapshotclasses/{name}**
  - **DELETE**: delete a VolumeSnapshotClass
  - **GET**: read the specified VolumeSnapshotClass
  - **PATCH**: partially update the specified VolumeSnapshotClass
  - **PUT**: replace the specified VolumeSnapshotClass

12.2.1. /apis/snapshot.storage.k8s.io/v1/volumesnapshotclasses

HTTP method

DELETE

Description

delete collection of VolumeSnapshotClass

Table 12.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind VolumeSnapshotClass

Table 12.2. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotClassList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a VolumeSnapshotClass

Table 12.3. 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.

Table 12.4. Body parameters

Parameter	Type	Description
body	**VolumeSnapshotClass** schema	

Table 12.5. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotClass** schema
201 - Created	**VolumeSnapshotClass** schema
202 - Accepted	**VolumeSnapshotClass** schema
401 - Unauthorized	Empty

12.2.2. /apis/snapshot.storage.k8s.io/v1/volumesnapshotclasses/{name}

Table 12.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeSnapshotClass

HTTP method

DELETE

Description

delete a VolumeSnapshotClass

Table 12.7. 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

Table 12.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified VolumeSnapshotClass

Table 12.9. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotClass** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified VolumeSnapshotClass

Table 12.10. 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.

Table 12.11. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotClass** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified VolumeSnapshotClass

Table 12.12. 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.

Table 12.13. Body parameters

Parameter	Type	Description
body	**VolumeSnapshotClass** schema	

Table 12.14. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotClass** schema
201 - Created	**VolumeSnapshotClass** schema
401 - Unauthorized	Empty

CHAPTER 13. VOLUMESNAPSHOTCONTENT [SNAPSHOT.STORAGE.K8S.IO/V1]

Description

VolumeSnapshotContent represents the actual "on-disk" snapshot object in the underlying storage system

Type

object

Required

- **spec**

13.1. 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
metadata	**ObjectMeta**	Standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	**object**	spec defines properties of a VolumeSnapshotContent created by the underlying storage system. Required.

Property	Type	Description
status	**object**	status represents the current information of a snapshot.

13.1.1. .spec

Description

spec defines properties of a VolumeSnapshotContent created by the underlying storage system. Required.

Type

object

Required

- **deletionPolicy**
- **driver**
- **source**
- **volumeSnapshotRef**

Property	Type	Description

Property	Type	Description
deletionPolicy	**string**	deletionPolicy determines whether this VolumeSnapshotContent and its physical snapshot on the underlying storage system should be deleted when its bound VolumeSnapshot is deleted. Supported values are "Retain" and "Delete". "Retain" means that the VolumeSnapshotContent and its physical snapshot on underlying storage system are kept. "Delete" means that the VolumeSnapshotContent and its physical snapshot on underlying storage system are deleted. For dynamically provisioned snapshots, this field will automatically be filled in by the CSI snapshotter sidecar with the "DeletionPolicy" field defined in the corresponding VolumeSnapshotClass. For pre-existing snapshots, users MUST specify this field when creating the VolumeSnapshotContent object. Required.
driver	**string**	driver is the name of the CSI driver used to create the physical snapshot on the underlying storage system. This MUST be the same as the name returned by the CSI GetPluginName() call for that driver. Required.
source	**object**	source specifies whether the snapshot is (or should be) dynamically provisioned or already exists, and just requires a Kubernetes object representation. This field is immutable after creation. Required.

Property	Type	Description
sourceVolumeMode	**string**	SourceVolumeMode is the mode of the volume whose snapshot is taken. Can be either “Filesystem” or “Block”. If not specified, it indicates the source volume’s mode is unknown. This field is immutable. This field is an alpha field.
volumeSnapshotClassName	**string**	name of the VolumeSnapshotClass from which this snapshot was (or will be) created. Note that after provisioning, the VolumeSnapshotClass may be deleted or recreated with different set of values, and as such, should not be referenced post-snapshot creation.
volumeSnapshotRef	**object**	volumeSnapshotRef specifies the VolumeSnapshot object to which this VolumeSnapshotContent object is bound. VolumeSnapshot.Spec.VolumeSnapshotContentName field must reference to this VolumeSnapshotContent’s name for the bidirectional binding to be valid. For a pre-existing VolumeSnapshotContent object, name and namespace of the VolumeSnapshot object MUST be provided for binding to happen. This field is immutable after creation. Required.

13.1.2. .spec.source

Description

source specifies whether the snapshot is (or should be) dynamically provisioned or already exists, and just requires a Kubernetes object representation. This field is immutable after creation. Required.

Type

object

Property	Type	Description

Property	Type	Description
snapshotHandle	**string**	snapshotHandle specifies the CSI "snapshot_id" of a pre-existing snapshot on the underlying storage system for which a Kubernetes object representation was (or should be) created. This field is immutable.
volumeHandle	**string**	volumeHandle specifies the CSI "volume_id" of the volume from which a snapshot should be dynamically taken from. This field is immutable.

13.1.3. .spec.volumeSnapshotRef

Description

volumeSnapshotRef specifies the VolumeSnapshot object to which this VolumeSnapshotContent object is bound. VolumeSnapshot.Spec.VolumeSnapshotContentName field must reference to this VolumeSnapshotContent's name for the bidirectional binding to be valid. For a pre-existing VolumeSnapshotContent object, name and namespace of the VolumeSnapshot object MUST be provided for binding to happen. This field is immutable after creation. Required.

Type

object

Property	Type	Description
apiVersion	**string**	API version of the referent.

Property	Type	Description
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
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

Property	Type	Description
uid	**string**	UID of the referent. More info: https://kubernetes.io/docs/concepts/overview/working-with-objects/names/#uids

13.1.4. .status

Description

status represents the current information of a snapshot.

Type

object

Property	Type	Description
creationTime	**integer**	creationTime is the timestamp when the point-in-time snapshot is taken by the underlying storage system. In dynamic snapshot creation case, this field will be filled in by the CSI snapshotter sidecar with the "creation_time" value returned from CSI "CreateSnapshot" gRPC call. For a pre-existing snapshot, this field will be filled with the "creation_time" value returned from the CSI "ListSnapshots" gRPC call if the driver supports it. If not specified, it indicates the creation time is unknown. The format of this field is a Unix nanoseconds time encoded as an int64. On Unix, the command **date +%s%N** returns the current time in nanoseconds since 1970-01-01 00:00:00 UTC.
error	**object**	error is the last observed error during snapshot creation, if any. Upon success after retry, this error field will be cleared.

Property	Type	Description
readyToUse	**boolean**	readyToUse indicates if a snapshot is ready to be used to restore a volume. In dynamic snapshot creation case, this field will be filled in by the CSI snapshotter sidecar with the "ready_to_use" value returned from CSI "CreateSnapshot" gRPC call. For a pre-existing snapshot, this field will be filled with the "ready_to_use" value returned from the CSI "ListSnapshots" gRPC call if the driver supports it, otherwise, this field will be set to "True". If not specified, it means the readiness of a snapshot is unknown.
restoreSize	**integer**	restoreSize represents the complete size of the snapshot in bytes. In dynamic snapshot creation case, this field will be filled in by the CSI snapshotter sidecar with the "size_bytes" value returned from CSI "CreateSnapshot" gRPC call. For a pre-existing snapshot, this field will be filled with the "size_bytes" value returned from the CSI "ListSnapshots" gRPC call if the driver supports it. When restoring a volume from this snapshot, the size of the volume MUST NOT be smaller than the restoreSize if it is specified, otherwise the restoration will fail. If not specified, it indicates that the size is unknown.
snapshotHandle	**string**	snapshotHandle is the CSI "snapshot_id" of a snapshot on the underlying storage system. If not specified, it indicates that dynamic snapshot creation has either failed or it is still in progress.
volumeGroupSnapshotHandle	**string**	VolumeGroupSnapshotHandle is the CSI "group_snapshot_id" of a group snapshot on the underlying storage system.

13.1.5. .status.error

Description

error is the last observed error during snapshot creation, if any. Upon success after retry, this error field will be cleared.

Type

object

Property	Type	Description
message	**string**	message is a string detailing the encountered error during snapshot creation if specified. NOTE: message may be logged, and it should not contain sensitive information.
time	**string**	time is the timestamp when the error was encountered.

13.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents**
  - **DELETE**: delete collection of VolumeSnapshotContent
  - **GET**: list objects of kind VolumeSnapshotContent
  - **POST**: create a VolumeSnapshotContent
- **/apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents/{name}**
  - **DELETE**: delete a VolumeSnapshotContent
  - **GET**: read the specified VolumeSnapshotContent
  - **PATCH**: partially update the specified VolumeSnapshotContent
  - **PUT**: replace the specified VolumeSnapshotContent
- **/apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents/{name}/status**
  - **GET**: read status of the specified VolumeSnapshotContent
  - **PATCH**: partially update status of the specified VolumeSnapshotContent
  - **PUT**: replace status of the specified VolumeSnapshotContent

13.2.1. /apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents

HTTP method

DELETE

Description

delete collection of VolumeSnapshotContent

Table 13.1. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

list objects of kind VolumeSnapshotContent

Table 13.2. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContentList** schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a VolumeSnapshotContent

Table 13.3. 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.

Table 13.4. Body parameters

Parameter	Type	Description
body	**VolumeSnapshotContent** schema	

Table 13.5. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
201 - Created	**VolumeSnapshotContent** schema
202 - Accepted	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty

13.2.2. /apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents/{name}

Table 13.6. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeSnapshotContent

HTTP method

DELETE

Description

delete a VolumeSnapshotContent

Table 13.7. 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

Table 13.8. HTTP responses

HTTP code	Reponse body
200 - OK	**Status** schema
202 - Accepted	**Status** schema
401 - Unauthorized	Empty

HTTP method

GET

Description

read the specified VolumeSnapshotContent

Table 13.9. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update the specified VolumeSnapshotContent

Table 13.10. 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.

Table 13.11. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace the specified VolumeSnapshotContent

Table 13.12. 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.

Table 13.13. Body parameters

Parameter	Type	Description
body	**VolumeSnapshotContent** schema	

Table 13.14. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
201 - Created	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty

13.2.3. /apis/snapshot.storage.k8s.io/v1/volumesnapshotcontents/{name}/status

Table 13.15. Global path parameters

Parameter	Type	Description
name	**string**	name of the VolumeSnapshotContent

HTTP method

GET

Description

read status of the specified VolumeSnapshotContent

Table 13.16. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty

HTTP method

PATCH

Description

partially update status of the specified VolumeSnapshotContent

Table 13.17. 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.

Table 13.18. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty

HTTP method

PUT

Description

replace status of the specified VolumeSnapshotContent

Table 13.19. 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.

Table 13.20. Body parameters

Parameter	Type	Description
body	**VolumeSnapshotContent** schema	

Table 13.21. HTTP responses

HTTP code	Reponse body
200 - OK	**VolumeSnapshotContent** schema
201 - Created	**VolumeSnapshotContent** schema
401 - Unauthorized	Empty