



OpenShift Container Platform 4.20

Project APIs

Reference guide for project APIs

OpenShift Container Platform 4.20 Project APIs

Reference guide for project 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 project API objects and their detailed specifications.

Table of Contents

CHAPTER 1. PROJECT APIS	3
1.1. PROJECT [PROJECT.OPENSIFT.IO/V1]	3
1.2. PROJECTREQUEST [PROJECT.OPENSIFT.IO/V1]	3
CHAPTER 2. PROJECT [PROJECT.OPENSIFT.IO/V1]	4
2.1. SPECIFICATION	4
2.1.1. .spec	5
2.1.2. .status	5
2.2. API ENDPOINTS	6
2.2.1. /apis/project.opensift.io/v1/projects	6
2.2.2. /apis/project.opensift.io/v1/watch/projects	8
2.2.3. /apis/project.opensift.io/v1/projects/{name}	8
2.2.4. /apis/project.opensift.io/v1/watch/projects/{name}	11
CHAPTER 3. PROJECTREQUEST [PROJECT.OPENSIFT.IO/V1]	13
3.1. SPECIFICATION	13
3.2. API ENDPOINTS	14
3.2.1. /apis/project.opensift.io/v1/projectrequests	14

CHAPTER 1. PROJECT APIS

1.1. PROJECT [PROJECT.OPENSIFT.IO/V1]

Description

Projects are the unit of isolation and collaboration in OpenShift. A project has one or more members, a quota on the resources that the project may consume, and the security controls on the resources in the project. Within a project, members may have different roles – project administrators can set membership, editors can create and manage the resources, and viewers can see but not access running containers. In a normal cluster project administrators are not able to alter their quotas – that is restricted to cluster administrators.

Listing or watching projects will return only projects the user has the reader role on.

An OpenShift project is an alternative representation of a Kubernetes namespace. Projects are exposed as editable to end users while namespaces are not. Direct creation of a project is typically restricted to administrators, while end users should use the requestproject resource.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

1.2. PROJECTREQUEST [PROJECT.OPENSIFT.IO/V1]

Description

ProjectRequest is the set of options necessary to fully qualify a project request

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

CHAPTER 2. PROJECT [PROJECT.OPENSIFT.IO/V1]

Description

Projects are the unit of isolation and collaboration in OpenShift. A project has one or more members, a quota on the resources that the project may consume, and the security controls on the resources in the project. Within a project, members may have different roles – project administrators can set membership, editors can create and manage the resources, and viewers can see but not access running containers. In a normal cluster project administrators are not able to alter their quotas – that is restricted to cluster administrators.

Listing or watching projects will return only projects the user has the reader role on.

An OpenShift project is an alternative representation of a Kubernetes namespace. Projects are exposed as editable to end users while namespaces are not. Direct creation of a project is typically restricted to administrators, while end users should use the requestproject resource.

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

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_v2	metadata is the standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata
spec	object	ProjectSpec describes the attributes on a Project
status	object	ProjectStatus is information about the current status of a Project

2.1.1. .spec

Description

ProjectSpec describes the attributes on a Project

Type

object

Property	Type	Description
finalizers	array (string)	Finalizers is an opaque list of values that must be empty to permanently remove object from storage

2.1.2. .status

Description

ProjectStatus is information about the current status of a Project

Type

object

Property	Type	Description
conditions	array (NamespaceCondition_v2)	Represents the latest available observations of the project current state.

Property	Type	Description
phase	string	Phase is the current lifecycle phase of the project Possible enum values: - "Active" means the namespace is available for use in the system - "Terminating" means the namespace is undergoing graceful termination

2.2. API ENDPOINTS

The following API endpoints are available:

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

2.2.1. /apis/project.openshift.io/v1/projects

HTTP method

GET

Description

list or watch objects of kind Project

Table 2.1. HTTP responses

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

HTTP method**POST****Description**

create a Project

Table 2.2. 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 2.3. Body parameters

Parameter	Type	Description
body	Project schema	

Table 2.4. HTTP responses

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

2.2.2. /apis/project.openshift.io/v1/watch/projects

HTTP method

GET

Description

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

Table 2.5. HTTP responses

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

2.2.3. /apis/project.openshift.io/v1/projects/{name}

Table 2.6. Global path parameters

Parameter	Type	Description
name	string	name of the Project

HTTP method

DELETE

Description

delete a Project

Table 2.7. 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 2.8. HTTP responses

HTTP code	Reponse body
200 - OK	Status_v6 schema
202 - Accepted	Status_v6 schema
401 - Unauthorized	Empty

HTTP method**GET****Description**

read the specified Project

Table 2.9. HTTP responses

HTTP code	Reponse body
200 - OK	Project schema
401 - Unauthorized	Empty

HTTP method**PATCH****Description**

partially update the specified Project

Table 2.10. 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

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 2.11. HTTP responses

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

HTTP method**PUT****Description**

replace the specified Project

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

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 2.13. Body parameters

Parameter	Type	Description
body	Project schema	

Table 2.14. HTTP responses

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

2.2.4. /apis/project.openshift.io/v1/watch/projects/{name}

Table 2.15. Global path parameters

Parameter	Type	Description
name	string	name of the Project

HTTP method

GET

Description

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

Table 2.16. HTTP responses

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

CHAPTER 3. PROJECTREQUEST [PROJECT.OPENSIFT.IO/V1]

Description

ProjectRequest is the set of options necessary to fully qualify a project request

Compatibility level 1: Stable within a major release for a minimum of 12 months or 3 minor releases (whichever is longer).

Type

object

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
description	string	Description is the description to apply to a project
displayName	string	DisplayName is the display name to apply to a project
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_v2	metadata is the standard object's metadata. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

3.2. API ENDPOINTS

The following API endpoints are available:

- **/apis/project.openshift.io/v1/projectrequests**
 - **GET**: list objects of kind ProjectRequest
 - **POST**: create a ProjectRequest

3.2.1. /apis/project.openshift.io/v1/projectrequests

HTTP method

GET

Description

list objects of kind ProjectRequest

Table 3.1. HTTP responses

HTTP code	Response body
200 - OK	Status_v6 schema
401 - Unauthorized	Empty

HTTP method

POST

Description

create a ProjectRequest

Table 3.2. 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 3.3. Body parameters

Parameter	Type	Description
body	ProjectRequest schema	

Table 3.4. HTTP responses

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