Namespace APIs

LimitRange [v1] Namespace [v1] ResourceQ

LimitRange [v1]

Description

LimitRange sets resource usage limits for each kind of resource in a Namespace.

Type

`object`

Specification

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

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

.spec

Description

LimitRangeSpec defines a min/max usage limit for resources that match on kind.

Type

`object`

Required

`limits`

Property	Type	Description
`limits`	`array`	Limits is the list of LimitRangeItem objects that are enforced.

.spec.limits

Description

Limits is the list of LimitRangeItem objects that are enforced.

Type

`array`

.spec.limits[]

Description

LimitRangeItem defines a min/max usage limit for any resource that matches on kind.

Type

`object`

Required

`type`

Property	Type	Description
`default`	`object`	Default resource requirement limit value by resource name if resource limit is omitted.
`defaultRequest`	`object`	DefaultRequest is the default resource requirement request value by resource name if resource request is omitted.
`max`	`object`	Max usage constraints on this kind by resource name.

Property	Type	Description
`maxLimitRequestRatio`	`object`	MaxLimitRequestRatio if specified, the named resource must have a request and limit that are both non-zero where limit divided by request is less than or equal to the enumerated value; this represents the max burst for the named resource.
`min`	`object`	Min usage constraints on this kind by resource name.
`type`	`string`	Type of resource that this limit applies to.

.spec.limits[].default

Description

Default resource requirement limit value by resource name if resource limit is omitted.

Type

`object`

.spec.limits[].defaultRequest

Description

DefaultRequest is the default resource requirement request value by resource name if resource request is omitted.

Type

`object`

.spec.limits[].max

Description

Max usage constraints on this kind by resource name.

Type

`object`

.spec.limits[].maxLimitRequestRatio

Description

MaxLimitRequestRatio if specified, the named resource must have a request and limit that are both non-zero where limit divided by request is less than or equal to the enumerated value; this represents the max burst for the named resource.

Type

`object`

.spec.limits[].min

Description

Min usage constraints on this kind by resource name.

Type

`object`

API Endpoints

The following API endpoints are available:

- `/kubernetes/{cluster}/api/v1/namespaces/{namespace}/limitranges`
  - `DELETE`: delete collection of LimitRange
  - `GET`: list objects of kind LimitRange
  - `POST`: create a new LimitRange

- `/kubernetes/{cluster}/api/v1/namespaces/{namespace}/limitranges/{name}`
  - `DELETE`: delete the specified LimitRange
  - `GET`: read the specified LimitRange
  - `PATCH`: partially update the specified LimitRange
  - `PUT`: replace the specified LimitRange

/kubernetes/{cluster}/api/v1/namespaces/{namespace}/limitranges

HTTP method

`DELETE`

Description

delete collection of LimitRange

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind LimitRange

HTTP responses

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

HTTP method

`POST`

Description

create a new LimitRange

Query parameters

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

Body parameters

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

HTTP responses

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

/kubernetes/{cluster}/api/v1/namespaces/{namespace}/limitranges/{name}

HTTP method

`DELETE`

Description

delete the specified LimitRange

Query parameters

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

HTTP responses

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

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`GET`

Description

read the specified LimitRange

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified LimitRange

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

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

HTTP responses

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

HTTP method

`PUT`

Description

replace the specified LimitRange

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

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

Body parameters

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

HTTP responses

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

Namespace [v1]

Description

Namespace provides a scope for Names. Use of multiple namespaces is optional.

Type

`object`

Specification

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

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

.spec

Description

NamespaceSpec describes the attributes on a Namespace.

Type

`object`

Property	Type	Description
`finalizers`	`array`	Finalizers is an opaque list of values that must be empty to permanently remove object from storage. More info:

Property	Type	Description
		https://kubernetes.io/docs/tasks/administer-cluster/namespaces/ ↗

.spec.finalizers

Description

Finalizers is an opaque list of values that must be empty to permanently remove object from storage. More info: https://kubernetes.io/docs/tasks/administer-cluster/namespaces/

Type

`array`

.spec.finalizers[]

Type

`string`

.status

Description

NamespaceStatus is information about the current status of a Namespace.

Type

`object`

Property	Type	Description
`conditions`	`array`	Represents the latest available observations of a namespace's current state.

Property	Type	Description
`phase`	`string`	Phase is the current lifecycle phase of the namespace. More info: https://kubernetes.io/docs/tasks/administer-cluster/namespaces/ Possible enum values: - `"Active"` means the namespace is available for use in the system - `"Terminating"` means the namespace is undergoing graceful termination

.status.conditions

Description

Represents the latest available observations of a namespace's current state.

Type

`array`

.status.conditions[]

Description

NamespaceCondition contains details about state of namespace.

Type

`object`

Required

`type` `status`

Property	Type	Description
`lastTransitionTime`	`string`	Time is a wrapper around time.Time which supports correct marshaling to YAML and JSON. Wrappers are provided for many of the factory methods that the time package offers.
`message`	`string`	Human-readable message indicating details about last transition.
`reason`	`string`	Unique, one-word, CamelCase reason for the condition's last transition.
`status`	`string`	Status of the condition, one of True, False, Unknown.
`type`	`string`	Type of namespace controller condition.

API Endpoints

The following API endpoints are available:

- `/kubernetes/{cluster}/api/v1/namespaces`
  - `DELETE`: delete collection of Namespace
  - `GET`: list objects of kind Namespace
  - `POST`: create a new Namespace
- `/kubernetes/{cluster}/api/v1/namespaces/{name}`

- `DELETE` : delete the specified Namespace
- `GET` : read the specified Namespace
- `PATCH` : partially update the specified Namespace
- `PUT` : replace the specified Namespace

- `/kubernetes/{cluster}/api/v1/namespaces/{name}/status`
  - `GET` : read status of the specified Namespace
  - `PATCH` : partially update status of the specified Namespace
  - `PUT` : replace status of the specified Namespace

/kubernetes/{cluster}/api/v1/namespaces

HTTP method

`DELETE`

Description

delete collection of Namespace

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind Namespace

HTTP responses

HTTP code	Response body
200 - OK	`NamespaceList` schema

HTTP code	Response body
401 - Unauthorized	Empty

HTTP method

`POST`

Description

create a new Namespace

Query parameters

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

Body parameters

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

HTTP responses

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

/kubernetes/{cluster}/api/v1/namespaces/{name}

HTTP method

`DELETE`

Description

delete the specified Namespace

Query parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`Status` schema

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

HTTP method

`GET`

Description

read the specified Namespace

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified Namespace

Query parameters

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

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

HTTP responses

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

HTTP method

`PUT`

Description

replace the specified Namespace

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

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

Body parameters

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

HTTP responses

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

/kubernetes/{cluster}/api/v1/namespaces/{name}/status

HTTP method

`GET`

Description

read status of the specified Namespace

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified Namespace

Query parameters

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

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

HTTP responses

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

HTTP method

`PUT`

Description

replace status of the specified Namespace

Query parameters

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

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

Body parameters

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

HTTP responses

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

ResourceQuota [v1]

Description

ResourceQuota sets aggregate quota restrictions enforced per namespace

Type

`object`

Specification

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

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

.spec

Description

ResourceQuotaSpec defines the desired hard limits to enforce for Quota.

Type

`object`

Property	Type	Description
`hard`	`object`	hard is the set of desired hard limits for each named resource. More info:

Property	Type	Description
		https://kubernetes.io/docs/concepts/policy/resource-quotas/ ↗
`scopeSelector`	`object`	A scope selector represents the AND of the selectors represented by the scoped-resource selector requirements.
`scopes`	`array`	A collection of filters that must match each object tracked by a quota. If not specified, the quota matches all objects.

.spec.hard

Description

hard is the set of desired hard limits for each named resource. More info: https://kubernetes.io/docs/concepts/policy/resource-quotas/

Type

`object`

.spec.scopeSelector

Description

A scope selector represents the AND of the selectors represented by the scoped-resource selector requirements.

Type

`object`

Property	Type	Description
`matchExpressions`	`array`	A list of scope selector requirements by scope of the resources.

.spec.scopeSelector.matchExpressions

Description

A list of scope selector requirements by scope of the resources.

Type

`array`

.spec.scopeSelector.matchExpressions[]

Description

A scoped-resource selector requirement is a selector that contains values, a scope name, and an operator that relates the scope name and values.

Type

`object`

Required

`scopeName` `operator`

Property	Type	Description
`operator`	`string`	Represents a scope's relationship to a set of values. Valid operators are In, NotIn, Exists, DoesNotExist. Possible enum values: - `"DoesNotExist"` - `"Exists"`

Property	Type	Description
		• `"In"` • `"NotIn"`
`scopeName`	`string`	The name of the scope that the selector applies to. Possible enum values: • `"BestEffort"` Match all pod objects that have best effort quality of service • `"CrossNamespacePodAffinity"` Match all pod objects that have cross-namespace pod (anti)affinity mentioned. • `"NotBestEffort"` Match all pod objects that do not have best effort quality of service • `"NotTerminating"` Match all pod objects where spec.activeDeadlineSeconds is nil • `"PriorityClass"` Match all pod objects that have priority class mentioned • `"Terminating"` Match all pod objects where spec.activeDeadlineSeconds >=0
`values`	`array`	An array of string values. If the operator is In or NotIn, the values array must be non-empty. If the operator is Exists or DoesNotExist, the values array must be empty. This array is replaced during a strategic merge patch.

.spec.scopeSelector.matchExpressions[].values

Description

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

Type

`array`

.spec.scopeSelector.matchExpressions[].values[]

Type

`string`

.spec.scopes

Description

A collection of filters that must match each object tracked by a quota. If not specified, the quota matches all objects.

Type

`array`

.spec.scopes[]

Type

`string`

.status

Description

ResourceQuotaStatus defines the enforced hard limits and observed use.

Type

`object`

Property	Type	Description
`hard`	`object`	Hard is the set of enforced hard limits for each named resource. More info: https://kubernetes.io/docs/concepts/policy/resource-quotas/ ↗
`used`	`object`	Used is the current observed total usage of the resource in the namespace.

.status.hard

Description

Hard is the set of enforced hard limits for each named resource. More info: https://kubernetes.io/docs/concepts/policy/resource-quotas/

Type

`object`

.status.used

Description

Used is the current observed total usage of the resource in the namespace.

Type

`object`

API Endpoints

The following API endpoints are available:

- `/kubernetes/{cluster}/api/v1/namespaces/{namespace}/resourcequotas`

- `DELETE`: delete collection of ResourceQuota
- `GET`: list objects of kind ResourceQuota
- `POST`: create a new ResourceQuota

- `/kubernetes/{cluster}/api/v1/namespaces/{namespace}/resourcequotas/{name}`
  - `DELETE`: delete the specified ResourceQuota
  - `GET`: read the specified ResourceQuota
  - `PATCH`: partially update the specified ResourceQuota
  - `PUT`: replace the specified ResourceQuota
- `/kubernetes/{cluster}/api/v1/namespaces/{namespace}/resourcequotas/{name}/status`
  - `GET`: read status of the specified ResourceQuota
  - `PATCH`: partially update status of the specified ResourceQuota
  - `PUT`: replace status of the specified ResourceQuota

/kubernetes/{cluster}/api/v1/namespaces/{namespace}/resourcequotas

HTTP method

`DELETE`

Description

delete collection of ResourceQuota

HTTP responses

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

HTTP method

`GET`

Description

list objects of kind ResourceQuota

HTTP responses

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

HTTP method

`POST`

Description

create a new ResourceQuota

Query parameters

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

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

Body parameters

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

HTTP responses

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

/kubernetes/{cluster}/api/v1/namespaces/{namespace}/resourcequotas/{name}

HTTP method

`DELETE`

Description

delete the specified ResourceQuota

Query parameters

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

HTTP responses

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

HTTP method

`GET`

Description

read the specified ResourceQuota

HTTP responses

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

HTTP method

`PATCH`

Description

partially update the specified ResourceQuota

Query parameters

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

HTTP responses

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

HTTP method

`PUT`

Description

replace the specified ResourceQuota

Query parameters

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

Body parameters

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

HTTP responses

HTTP code	Response body
200 - OK	`ResourceQuota` schema

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

/kubernetes/{cluster}/api/v1/namespaces/{namespace}/resourcequotas/{name}/status

HTTP method

`GET`

Description

read status of the specified ResourceQuota

HTTP responses

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

HTTP method

`PATCH`

Description

partially update status of the specified ResourceQuota

Query parameters

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

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

HTTP responses

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

HTTP method

`PUT`

Description

replace status of the specified ResourceQuota

Query parameters

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

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

Body parameters

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

HTTP responses

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