Red Hat Quay 3.6

Red Hat Quay API Guide

Red Hat Quay API Guide

Last Updated: 2022-07-05

Red Hat Quay 3.6 Red Hat Quay API Guide

Red Hat Quay API Guide

Legal Notice

Abstract

Use the Red Hat Quay API

Table of Contents

PREFACE

The Red Hat Quay application programming interface (API) is an OAuth 2 RESTful API that consists of a set of endpoints for adding, displaying, changing and deleting features for Red Hat Quay. This guide describes those endpoints and shows command and browser-based examples for accessing them.

CHAPTER 1. USING THE RED HAT QUAY API

Red Hat Quay provides a full OAuth 2, RESTful API that:

- Is available from endpoints of each Red Hat Quay instance from the URL https://<yourquayhost>/api/v1
- Lets you connect to endpoints, via a browser, to get, delete, post, and put Red Hat Quay settings by enabling the Swagger UI
- Can be accessed by applications that make API calls and use OAuth tokens
- Sends and receives data as JSON

The following text describes how to access the Red Hat Quay API and use it to view and modify setting in your Red Hat Quay cluster. The next section lists and describes API endpoints.

1.1. ACCESSING THE QUAY API FROM QUAY.IO

If you don't have your own Red Hat Quay cluster running yet, you can explore the Red Hat Quay API available from Quay.io from your web browser:

> https://docs.quay.io/api/swagger/

The API Explorer that appears shows Quay.io API endpoints. You will not see superuser API endpoints or endpoints for Red Hat Quay features that are not enabled on Quay.io (such as Repository Mirroring).

From API Explorer, you can get, and sometimes change, information on:

- Billing, subscriptions, and plans
- Repository builds and build triggers
- Error messages and global messages
- Repository images, manifests, permissions, notifications, vulnerabilities, and image signing
- Usage logs
- Organizations, members and OAuth applications
- User and robot accounts
- and more...

Select to open an endpoint to view the Model Schema for each part of the endpoint. Open an endpoint, enter any required parameters (such as a repository name or image), then select the **Try it out!** button to query or change settings associated with a Quay.io endpoint.

1.2. CREATE OAUTH ACCESS TOKEN

To create an OAuth access token so you can access the API for your organization:

1. Log in to Red Hat Quay and select your Organization (or create a new one).

2. Select the Applications icon from the left navigation.
3. Select Create New Application and give the new application a name when prompted.
4. Select the new application.
5. Select Generate Token from the left navigation.
6. Select the checkboxes to set the scope of the token and select Generate Access Token.
7. Review the permissions you are allowing and select Authorize Application to approve it.
8. Copy the newly generated token to use to access the API.

1.3. ACCESSING YOUR QUAY API FROM A WEB BROWSER

By enabling Swagger, you can access the API for your own Red Hat Quay instance through a web browser. This URL exposes the Red Hat Quay API explorer via the Swagger UI and this URL:

```
https://<yourquayhost>/api/v1/discovery.
```

That way of accessing the API does not include superuser endpoints that are available on Red Hat Quay installations. Here is an example of accessing a Red Hat Quay API interface running on the local system by running the swagger-ui container image:

```
# export SERVER_HOSTNAME=<yourhostname>
# sudo podman run -p 8888:8080 -e API_URL=https://$SERVER_HOSTNAME:8443/api/v1/discovery
docker.io/swaggerapi/swagger-ui
```

With the swagger-ui container running, open your web browser to localhost port 8888 to view API endpoints via the swagger-ui container.

To avoid errors in the log such as "API calls must be invoked with an X-Requested-With header if called from a browser," add the following line to the **config.yaml** on all nodes in the cluster and restart Red Hat Quay:

```
BROWSER_API_CALLS_XHR_ONLY: false
```

1.4. ACCESSING THE RED HAT QUAY API FROM THE COMMAND LINE

You can use the **curl** command to GET, PUT, POST, or DELETE settings via the API for your Red Hat Quay cluster. Replace **<token>** with the OAuth access token you created earlier to get or change settings in the following examples.

1.4.1. Get superuser information

```
$ curl -X GET -H "Authorization: Bearer <token_here>" \
    "https://<yourquayhost>/api/v1/superuser/users/"
```

For example:

```
$ curl -X GET -H "Authorization: Bearer mFCdgS7SAIoMcnTsHCGx23vcNsTgziAa4CmmHIsg"
http://quay-server:8080/api/v1/superuser/users/ | jq
```

```
{
  "users": [
    {
      "kind": "user",
      "name": "quayadmin",
      "username": "quayadmin",
      "email": "quayadmin@example.com",
      "verified": true,
      "avatar": {
        "name": "quayadmin",
        "hash": "357a20e8c56e69d6f9734d23ef9517e8",
        "color": "#5254a3",
        "kind": "user"
      },
      "super_user": true,
      "enabled": true
    }
  ]
}
```


1.4.2. Creating a superuser using the API

- Configure a superuser name, as described in the Deploy Quay book:
  - Use the configuration editor UI or
  - Edit the **config.yaml** file directly, with the option of using the configuration API to validate (and download) the updated configuration bundle
- Create the user account for the superuser name:
  - Obtain an authorization token as detailed above, and use **curl** to create the user:

```
$ curl -H "Content-Type: application/json"  -H "Authorization: Bearer
Fava2kV9C92p1eXnMawBZx9vTqVnksvwNm0ckFKZ" -X POST --data '{
 "username": "quaysuper",
 "email": "quaysuper@example.com"
}'  http://quay-server:8080/api/v1/superuser/users/ | jq
```

  - The returned content includes a generated password for the new user account:


```
{
  "username": "quaysuper",
  "email": "quaysuper@example.com",
  "password": "EH67NB3Y6PTBED8H0HC6UVHGGGA3ODSE",
  "encrypted_password":
"fn37AZAUQH0PTsU+vlO9lS0QxPW9A/boXL4ovZjIFtlUPrBz9i4j9UDOqMjuxQ/0HTfy38go
KEpG8zYXVeQh3lOFzuOjSvKic2Vq7xdtQsU="
}
```


Now, when you request the list of users , it will show **quaysuper** as a superuser:

```
$ curl -X GET -H "Authorization: Bearer mFCdgS7SAIoMcnTsHCGx23vcNsTgziAa4CmmHIsg"
http://quay-server:8080/api/v1/superuser/users/ | jq
```

```
{
  "users": [
   {
      "kind": "user",
      "name": "quayadmin",
      "username": "quayadmin",
      "email": "quayadmin@example.com",
      "verified": true,
      "avatar": {
        "name": "quayadmin",
        "hash": "357a20e8c56e69d6f9734d23ef9517e8",
        "color": "#5254a3",
        "kind": "user"
      },
      "super_user": true,
      "enabled": true
    },
    {
      "kind": "user",
      "name": "quaysuper",
      "username": "quaysuper",
      "email": "quaysuper@example.com",
      "verified": true,
      "avatar": {
        "name": "quaysuper",
        "hash": "c0e0f155afcef68e58a42243b153df08",
        "color": "#969696",
        "kind": "user"
      },
      "super_user": true,
      "enabled": true
    }
  ]
}
```


1.4.3. List usage logs

An intrnal API, **/api/v1/superuser/logs**, is available to list the usage logs for the current system. The results are paginated, so in the following example, more than 20 repos were created to show how to use multiple invocations to access the entire result set.

1.4.3.1. Example for pagination

First invocation

```
$ curl -X GET -k -H "Authorization: Bearer qz9NZ2Np1f55CSZ3RVOvxjeUdkzYuCp0pKggABCD"
https://example-registry-quay-quay-enterprise.apps.example.com/api/v1/superuser/logs | jq
```

Initial output


```
{
  "start_time": "Sun, 12 Dec 2021 11:41:55 -0000",
  "end_time": "Tue, 14 Dec 2021 11:41:55 -0000",
```

```
"logs": [
  {
    "kind": "create_repo",
    "metadata": {
      "repo": "t21",
      "namespace": "namespace1"
    },
    "ip": "10.131.0.13",
    "datetime": "Mon, 13 Dec 2021 11:41:16 -0000",
    "performer": {
      "kind": "user",
      "name": "user1",
      "is_robot": false,
      "avatar": {
        "name": "user1",
        "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
        "color": "#ad494a",
        "kind": "user"
      }
    },
    "namespace": {
      "kind": "org",
      "name": "namespace1",
      "avatar": {
        "name": "namespace1",
        "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
        "color": "#e377c2",
        "kind": "org"
      }
    }
  },
  {
    "kind": "create_repo",
    "metadata": {
      "repo": "t20",
      "namespace": "namespace1"
    },
    "ip": "10.131.0.13",
    "datetime": "Mon, 13 Dec 2021 11:41:05 -0000",
    "performer": {
      "kind": "user",
      "name": "user1",
      "is_robot": false,
      "avatar": {
        "name": "user1",
        "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
        "color": "#ad494a",
        "kind": "user"
      }
    },
    "namespace": {
      "kind": "org",
      "name": "namespace1",
      "avatar": {
        "name": "namespace1",
        "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
```

```
          "color": "#e377c2",
          "kind": "org"
        }
      }
    },
...

  {
    "kind": "create_repo",
    "metadata": {
      "repo": "t2",
      "namespace": "namespace1"
    },
    "ip": "10.131.0.13",
    "datetime": "Mon, 13 Dec 2021 11:25:17 -0000",
    "performer": {
      "kind": "user",
      "name": "user1",
      "is_robot": false,
      "avatar": {
        "name": "user1",
        "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
        "color": "#ad494a",
        "kind": "user"
      }
    },
    "namespace": {
      "kind": "org",
      "name": "namespace1",
      "avatar": {
        "name": "namespace1",
        "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
        "color": "#e377c2",
        "kind": "org"
      }
    }
  }
 ],
  "next_page":
"gAAAAABhtzGDsH38x7pjWhD8MJq1_2FAgqUw2X9S2LoCLNPH65QJqB4XAU2qAxYb6QqtlcWj9eI6
DUiMN_q3e3I0agCvB2VPQ8rY75WeaiUzM3rQlMc4i6ElR78t8oUxVfNp1RMPIRQYYZyXP9h6E8LZZhq
TMs0S-SedaQJ3kVFtkxZqJwHVjgt23Ts2DonVoYwtKgI3bCC5"
}
```


Second invocation using next_page

```
$ curl -X GET -k -H "Authorization: Bearer qz9NZ2Np1f55CSZ3RVOvxjeUdkzYuCp0pKggABCD"
https://example-registry-quay-quay-enterprise.apps.example.com/api/v1/superuser/logs?
next_page=gAAAAABhtzGDsH38x7pjWhD8MJq1_2FAgqUw2X9S2LoCLNPH65QJqB4XAU2qAxYb6Q
qtlcWj9eI6DUiMN_q3e3I0agCvB2VPQ8rY75WeaiUzM3rQlMc4i6ElR78t8oUxVfNp1RMPIRQYYZyXP9h
6E8LZZhqTMs0S-SedaQJ3kVFtkxZqJwHVjgt23Ts2DonVoYwtKgI3bCC5 | jq
```

Output from second invocation


```
{
```

```
  "start_time": "Sun, 12 Dec 2021 11:42:46 -0000",
  "end_time": "Tue, 14 Dec 2021 11:42:46 -0000",
  "logs": [
    {
      "kind": "create_repo",
      "metadata": {
        "repo": "t1",
        "namespace": "namespace1"
      },
      "ip": "10.131.0.13",
      "datetime": "Mon, 13 Dec 2021 11:25:07 -0000",
      "performer": {
        "kind": "user",
        "name": "user1",
        "is_robot": false,
        "avatar": {
          "name": "user1",
          "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
          "color": "#ad494a",
          "kind": "user"
        }
      },
      "namespace": {
        "kind": "org",
        "name": "namespace1",
        "avatar": {
          "name": "namespace1",
          "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
          "color": "#e377c2",
          "kind": "org"
        }
      }
    },
    ...
  ]
}
```


1.4.4. Directory synchronization

To enable directory synchronization for the team **newteam** in organization **testadminorg**, where the corresponding group name in LDAP is **ldapgroup**:

```
$ curl -X POST -H "Authorization: Bearer 9rJYBR3v3pXcj5XqIA2XX6Thkwk4gld4TCYLLWDF" \
      -H "Content-type: application/json" \
      -d '{"group_dn": "cn=ldapgroup,ou=Users"}' \
      http://quay1-server:8080/api/v1/organization/testadminorg/team/newteam/syncing
```

To disable synchronization for the same team:

```
$ curl -X DELETE -H "Authorization: Bearer 9rJYBR3v3pXcj5XqIA2XX6Thkwk4gld4TCYLLWDF" \
      http://quay1-server:8080/api/v1/organization/testadminorg/team/newteam/syncing
```

1.4.5. Create a repository build via API

In order to build a repository from the specified input and tag the build with custom tags, users can use requestRepoBuild endpoint. It takes the following data:

```
{
"docker_tags": [
  "string"
],
"pull_robot": "string",
"subdirectory": "string",
"archive_url": "string"
}
```

The **archive_url** parameter should point to a **tar** or **zip** archive that includes the Dockerfile and other required files for the build. The **file_id** parameter was apart of our older build system. It cannot be used anymore. If Dockerfile is in a sub-directory it needs to be specified as well.

The archive should be publicly accessible. OAuth app should have "Administer Organization" scope because only organization admins have access to the robots' account tokens. Otherwise, someone could get robot permissions by simply granting a build access to a robot (without having access themselves), and use it to grab the image contents. In case of errors, check the json block returned and ensure the archive location, pull robot, and other parameters are being passed correctly. Click "Download logs" on the top-right of the individual build's page to check the logs for more verbose messaging.

1.4.6. Create an org robot

```
$ curl -X PUT https://quay.io/api/v1/organization/{orgname}/robots/{robot shortname} \
  -H 'Authorization: Bearer <token>"
```

1.4.7. Trigger a build

```
$ curl -X POST https://quay.io/api/v1/repository/YOURORGNAME/YOURREPONAME/build/ \
  -H 'Authorization: Bearer <token>'
```

Python with requests

```
import requests
r = requests.post('https://quay.io/api/v1/repository/example/example/image', headers={'content-type':
'application/json', 'Authorization': 'Bearer <redacted>'}, data={[<request-body-contents>})
print(r.text)
```

1.4.8. Create a private repository

```
$ curl -X POST https://quay.io/api/v1/repository \
   -H 'Authorization: Bearer {token}' \
   -H 'Content-Type: application/json' \
   -d '{"namespace":"yournamespace", "repository":"yourreponame",
   "description":"descriptionofyourrepo", "visibility": "private"}' | jq
```

CHAPTER 2. RED HAT QUAY APPLICATION PROGRAMMING INTERFACE (API)

This API allows you to perform many of the operations required to work with Red Hat Quay repositories, users, and organizations.

2.1. AUTHORIZATION

oauth2_implicit

Scopes

The following scopes are used to control access to the API endpoints:

Scope	Description
repo:read	This application will be able to view and pull all repositories visible to the granting user or robot account
repo:write	This application will be able to view, push and pull to all repositories to which the granting user or robot account has write access
repo:admin	This application will have administrator access to all repositories to which the granting user or robot account has access
repo:create	This application will be able to create repositories in to any namespaces that the granting user or robot account is allowed to create repositories
user:read	This application will be able to read user information such as username and email address.
org:admin	This application will be able to administer your organizations including creating robots, creating teams, adjusting team membership, and changing billing settings. You should have absolute trust in the requesting application before granting this permission.
super:user	This application will be able to administer your installation including managing users, managing organizations and other features found in the superuser panel. You should have absolute trust in the requesting application before granting this permission.
user:admin	This application will be able to administer your account including creating robots and granting them permissions to your repositories. You should have absolute trust in the requesting application before granting this permission.

2.2. APPSPECIFICTOKENS

Manages app specific tokens for the current user.

2.2.1. listAppTokens

Lists the app specific tokens for the user.

GET /api/v1/user/apptoken

Authorizations: oauth2_implicit (**user:admin**)

Query parameters

Type	Name	Description	Schema
query	**expiring** *optional*	If true, only returns those tokens expiring soon	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.2.2. createAppToken

Create a new app specific token for user.

POST /api/v1/user/apptoken
Authorizations: oauth2_implicit (**user:admin**)

Request body schema (application/json)
Description of a new token.

Name	Description	Schema
friendlyName *optional*	Friendly name to help identify the token	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.2.3. revokeAppToken

Revokes a specific app token for the user.

DELETE /api/v1/user/apptoken/{token_uuid}
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**token_uuid** *required*	The uuid of the app specific token	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.2.4. getAppToken

Returns a specific app token for the user.

GET /api/v1/user/apptoken/{token_uuid}
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**token_uuid** *required*	The uuid of the app specific token	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.3. BUILD

Create, list, cancel and get status/logs of repository builds.

2.3.1. getRepoBuildStatus

Return the status for the builds specified by the build uuids.

GET /api/v1/repository/{repository}/build/{build_uuid}/status
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.3.2. getRepoBuildLogs

Return the build logs for the build specified by the build uuid.

GET /api/v1/repository/{repository}/build/{build_uuid}/logs
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.3.3. cancelRepoBuild

Cancels a repository build.

DELETE /api/v1/repository/{repository}/build/{build_uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.3.4. getRepoBuild

Returns information about a build.

GET /api/v1/repository/{repository}/build/{build_uuid}
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.3.5. getRepoBuilds

Get the list of repository builds.

GET /api/v1/repository/{repository}/build/
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema
query	**since** *optional*	Returns all builds since the given unix timecode	integer
query	**limit** *optional*	The maximum number of builds to return	integer

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.3.6. requestRepoBuild

Request that a repository be built and pushed from the specified input.

POST /api/v1/repository/{repository}/build/
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)

Description of a new repository build.

Name	Description	Schema
file_id *optional*	The file id that was generated when the build spec was uploaded	string
archive_url *optional*	The URL of the .tar.gz to build. Must start with "http" or "https".	string
subdirectory *optional*	Subdirectory in which the Dockerfile can be found. You can only specify this or dockerfile_path	string
dockerfile_path *optional*	Path to a dockerfile. You can only specify this or subdirectory.	string
context *optional*	Pass in the context for the dockerfile. This is optional.	string
pull_robot *optional*	Username of a Quay robot account to use as pull credentials	string
docker_tags *optional*	The tags to which the built images will be pushed. If none specified, "latest" is used.	array of string **non-empty unique**

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.4. DISCOVERY

API discovery information.

2.4.1. discovery

List all of the API endpoints available in the swagger API format.

GET /api/v1/discovery

Authorizations:

Query parameters

Type	Name	Description	Schema
query	**internal** *optional*	Whether to include internal APIs.	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.5. ERROR

Error details API.

2.5.1. getErrorDescription

Get a detailed description of the error.

GET /api/v1/error/{error_type}
Authorizations:

Path parameters

Type	Name	Description	Schema
path	**error_type** *required*	The error code identifying the type of error.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	ApiErrorDescription

HTTP Code	Description	Schema
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.6. GLOBALMESSAGES

Messages API.

2.6.1. getGlobalMessages

Return a super users messages.

GET /api/v1/messages
Authorizations:

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.6.2. createGlobalMessage

Create a message.

POST /api/v1/messages
Authorizations: oauth2_implicit (**super:user**)

Request body schema (application/json)
Create a new message

Name	Description	Schema
message *optional*	A single message	object

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.6.3. deleteGlobalMessage

Delete a message.

DELETE /api/v1/message/{uuid}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**uuid** *required*		string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.7. IMAGE

List and lookup repository images.

2.7.1. getImage

Get the information available for the specified image.

GET /api/v1/repository/{repository}/image/{image_id}
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**image_id** *required*	The Docker image ID	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.7.2. listRepositoryImages

List the images for the specified repository.

GET /api/v1/repository/{repository}/image/
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8. LOGS

Access usage logs for organizations or repositories.

2.8.1. getAggregateUserLogs

Returns the aggregated logs for the current user.

GET /api/v1/user/aggregatelogs
Authorizations: oauth2_implicit (**user:admin**)

Query parameters

Type	Name	Description	Schema
query	**performer** *optional*	Username for which to filter logs.	string
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError

HTTP Code	Description	Schema
404	Not found	ApiError

2.8.2. exportUserLogs

Returns the aggregated logs for the current user.

POST /api/v1/user/exportlogs
Authorizations: oauth2_implicit (**user:admin**)

Query parameters

Type	Name	Description	Schema
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Request body schema (application/json)
Configuration for an export logs operation

Name	Description	Schema
callback_url *optional*	The callback URL to invoke with a link to the exported logs	string
callback_email *optional*	The e-mail address at which to e-mail a link to the exported logs	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.3. listUserLogs

List the logs for the current user.

GET /api/v1/user/logs
Authorizations: oauth2_implicit (**user:admin**)

Query parameters

Type	Name	Description	Schema
query	**next_page** *optional*	The page token for the next page	string
query	**performer** *optional*	Username for which to filter logs.	string
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.4. getAggregateOrgLogs

Gets the aggregated logs for the specified organization.

GET /api/v1/organization/{orgname}/aggregatelogs
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Query parameters

Type	Name	Description	Schema
query	**performer** *optional*	Username for which to filter logs.	string
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.5. exportOrgLogs

Exports the logs for the specified organization.

POST /api/v1/organization/{orgname}/exportlogs
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Query parameters

Type	Name	Description	Schema
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string

Type	Name	Description	Schema
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Request body schema (application/json)
Configuration for an export logs operation

Name	Description	Schema
callback_url *optional*	The callback URL to invoke with a link to the exported logs	string
callback_email *optional*	The e-mail address at which to e-mail a link to the exported logs	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.6. listOrgLogs

List the logs for the specified organization.

GET /api/v1/organization/{orgname}/logs
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Query parameters

Type	Name	Description	Schema
query	**next_page** *optional*	The page token for the next page	string
query	**performer** *optional*	Username for which to filter logs.	string
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.7. getAggregateRepoLogs

Returns the aggregated logs for the specified repository.

GET /api/v1/repository/{repository}/aggregatelogs
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string

Type	Name	Description	Schema
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.8. exportRepoLogs

Queues an export of the logs for the specified repository.

POST /api/v1/repository/{repository}/exportlogs
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Request body schema (application/json)
Configuration for an export logs operation

Name	Description	Schema
callback_url *optional*	The callback URL to invoke with a link to the exported logs	string
callback_email *optional*	The e-mail address at which to e-mail a link to the exported logs	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.8.9. listRepoLogs

List the logs for the specified repository.

GET /api/v1/repository/{repository}/logs
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema
query	**next_page** *optional*	The page token for the next page	string
query	**endtime** *optional*	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	**starttime** *optional*	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.9. MANIFEST

Manage the manifests of a repository.

2.9.1. deleteManifestLabel

Deletes an existing label from a manifest.

DELETE /api/v1/repository/{repository}/manifest/{manifestref}/labels/{labelid}
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**labelid** *required*	The ID of the label	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**manifestref** *required*	The digest of the manifest	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.9.2. getManifestLabel

Retrieves the label with the specific ID under the manifest.

GET /api/v1/repository/{repository}/manifest/{manifestref}/labels/{labelid}
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**labelid** *required*	The ID of the label	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**manifestref** *required*	The digest of the manifest	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.9.3. listManifestLabels

GET /api/v1/repository/{repository}/manifest/{manifestref}/labels
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**manifestref** *required*	The digest of the manifest	string

Query parameters

Type	Name	Description	Schema
query	**filter** *optional*	If specified, only labels matching the given prefix will be returned	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.9.4. addManifestLabel

Adds a new label into the tag manifest.

POST /api/v1/repository/{repository}/manifest/{manifestref}/labels
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**manifestref** *required*	The digest of the manifest	string

Request body schema (application/json)
Adds a label to a manifest

Name	Description	Schema
key *optional*	The key for the label	string
value *optional*	The value for the label	string
media_type *optional*	The media type for this label	

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.9.5. getRepoManifest

GET /api/v1/repository/{repository}/manifest/{manifestref}
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**manifestref** *required*	The digest of the manifest	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.10. MIRROR

2.10.1. syncCancel

Update the sync_status for a given Repository's mirroring configuration.

POST /api/v1/repository/{repository}/mirror/sync-cancel
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.10.2. syncNow

Update the sync_status for a given Repository's mirroring configuration.

POST /api/v1/repository/{repository}/mirror/sync-now
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.10.3. getRepoMirrorConfig

Return the Mirror configuration for a given Repository.

GET /api/v1/repository/{repository}/mirror
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	ViewMirrorConfig
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.10.4. createRepoMirrorConfig

Create a RepoMirrorConfig for a given Repository.

POST /api/v1/repository/{repository}/mirror
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Create the repository mirroring configuration.

Name	Description	Schema
is_enabled *optional*	Used to enable or disable synchronizations.	boolean
external_reference *optional*	Location of the external repository.	string
external_registry_username *optional*	Username used to authenticate with external registry.	
external_registry_password *optional*	Password used to authenticate with external registry.	
sync_start_date *optional*	Determines the next time this repository is ready for synchronization.	string
sync_interval *optional*	Number of seconds after next_start_date to begin synchronizing.	integer
robot_username *optional*	Username of robot which will be used for image pushes.	string
root_rule *optional*	A list of glob-patterns used to determine which tags should be synchronized.	object
external_registry_config *optional*		object

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.10.5. changeRepoMirrorConfig

Allow users to modifying the repository's mirroring configuration.

PUT /api/v1/repository/{repository}/mirror
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Update the repository mirroring configuration.

Name	Description	Schema
is_enabled *optional*	Used to enable or disable synchronizations.	boolean
external_reference *optional*	Location of the external repository.	string
external_registry_username *optional*	Username used to authenticate with external registry.	
external_registry_password *optional*	Password used to authenticate with external registry.	
sync_start_date *optional*	Determines the next time this repository is ready for synchronization.	string

Name	Description	Schema
sync_interval *optional*	Number of seconds after next_start_date to begin synchronizing.	integer
robot_username *optional*	Username of robot which will be used for image pushes.	string
root_rule *optional*	A list of glob-patterns used to determine which tags should be synchronized.	object
external_registry_config *optional*		object

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11. ORGANIZATION

Manage organizations, members and OAuth applications.

2.11.1. createOrganization

Create a new organization.

POST /api/v1/organization/
Authorizations: oauth2_implicit (**user:admin**)

Request body schema (application/json)
Description of a new organization.

Name	Description	Schema
name *optional*	Organization username	string

Name	Description	Schema
email *optional*	Organization contact email	string
recaptcha_response *optional*	The (may be disabled) recaptcha response code for verification	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.2. getOrganizationCollaborators

List outside collaborators of the specified organization.

GET /api/v1/organization/{orgname}/collaborators
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.3. deleteOrganizationApplication

Deletes the application under this organization.

DELETE /api/v1/organization/{orgname}/applications/{client_id}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**client_id** *required*	The OAuth client ID	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.4. getOrganizationApplication

Retrieves the application with the specified client_id under the specified organization.

GET /api/v1/organization/{orgname}/applications/{client_id}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**client_id** *required*	The OAuth client ID	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.5. updateOrganizationApplication

Updates an application under this organization.

PUT /api/v1/organization/{orgname}/applications/{client_id}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**client_id** *required*	The OAuth client ID	string

Request body schema (application/json)
Description of an updated application.

Name	Description	Schema
name *optional*	The name of the application	string

Name	Description	Schema
redirect_uri *optional*	The URI for the application's OAuth redirect	string
application_uri *optional*	The URI for the application's homepage	string
description *optional*	The human-readable description for the application	string
avatar_email *optional*	The e-mail address of the avatar to use for the application	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.6. getOrganizationApplications

List the applications for the specified organization.

GET /api/v1/organization/{orgname}/applications
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.7. createOrganizationApplication

Creates a new application under this organization.

POST /api/v1/organization/{orgname}/applications
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Request body schema (application/json)
Description of a new organization application.

Name	Description	Schema
name *optional*	The name of the application	string
redirect_uri *optional*	The URI for the application's OAuth redirect	string
application_uri *optional*	The URI for the application's homepage	string
description *optional*	The human-readable description for the application	string
avatar_email *optional*	The e-mail address of the avatar to use for the application	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.8. removeOrganizationMember

Removes a member from an organization, revoking all its repository priviledges and removing it from all teams in the organization.

DELETE /api/v1/organization/{orgname}/members/{membername}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**membername** *required*	The username of the organization member	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.9. getOrganizationMember

Retrieves the details of a member of the organization.

GET /api/v1/organization/{orgname}/members/{membername}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**membername** *required*	The username of the organization member	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.10. getOrganizationMembers

List the human members of the specified organization.

GET /api/v1/organization/{orgname}/members
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.11. deleteAdminedOrganization

Deletes the specified organization.

DELETE /api/v1/organization/{orgname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.12. getOrganization

Get the details for the specified organization.

GET /api/v1/organization/{orgname}
Authorizations:

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.13. changeOrganizationDetails

Change the details for the specified organization.

PUT /api/v1/organization/{orgname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Request body schema (application/json)
Description of updates for an existing organization

Name	Description	Schema
email *optional*	Organization contact email	string
invoice_email *optional*	Whether the organization desires to receive emails for invoices	boolean

Name	Description	Schema
invoice_email_address *optional*	The email address at which to receive invoices	
tag_expiration_s *optional*	The number of seconds for tag expiration	integer

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.11.14. getApplicationInformation

Get information on the specified application.

GET /api/v1/app/{client_id}
Authorizations:

Path parameters

Type	Name	Description	Schema
path	**client_id** *required*	The OAuth client ID	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12. PERMISSION

Manage repository permissions.

2.12.1. getUserTransitivePermission

Get the fetch the permission for the specified user.

GET /api/v1/repository/{repository}/permissions/user/{username}/transitive
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**username** *required*	The username of the user to which the permissions apply	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.2. deleteUserPermissions

Delete the permission for the user.

DELETE /api/v1/repository/{repository}/permissions/user/{username}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**username** *required*	The username of the user to which the permission applies	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.3. getUserPermissions

Get the permission for the specified user.

GET /api/v1/repository/{repository}/permissions/user/{username}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**username** *required*	The username of the user to which the permission applies	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.4. changeUserPermissions

Update the perimssions for an existing repository.

PUT /api/v1/repository/{repository}/permissions/user/{username}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**username** *required*	The username of the user to which the permission applies	string

Request body schema (application/json)
Description of a user permission.

Name	Description	Schema
role *optional*	Role to use for the user	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.5. deleteTeamPermissions

Delete the permission for the specified team.

DELETE /api/v1/repository/{repository}/permissions/team/{teamname}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**teamname** *required*	The name of the team to which the permission applies	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.6. getTeamPermissions

Fetch the permission for the specified team.

GET /api/v1/repository/{repository}/permissions/team/{teamname}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**teamname** *required*	The name of the team to which the permission applies	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.7. changeTeamPermissions

Update the existing team permission.

PUT /api/v1/repository/{repository}/permissions/team/{teamname}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**teamname** *required*	The name of the team to which the permission applies	string

Request body schema (application/json)
Description of a team permission.

Name	Description	Schema
role *optional*	Role to use for the team	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.8. listRepoTeamPermissions

List all team permission.

GET /api/v1/repository/{repository}/permissions/team/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.12.9. listRepoUserPermissions

List all user permissions.

GET /api/v1/repository/{repository}/permissions/user/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.13. PROTOTYPE

Manage default permissions added to repositories.

2.13.1. deleteOrganizationPrototypePermission

Delete an existing permission prototype.

DELETE /api/v1/organization/{orgname}/prototypes/{prototypeid}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**prototypeid** *required*	The ID of the prototype	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.13.2. updateOrganizationPrototypePermission

Update the role of an existing permission prototype.

PUT /api/v1/organization/{orgname}/prototypes/{prototypeid}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**prototypeid** *required*	The ID of the prototype	string

Request body schema (application/json)
Description of a the new prototype role

Name	Description	Schema
role *optional*	Role that should be applied to the permission	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.13.3. getOrganizationPrototypePermissions

List the existing prototypes for this organization.

GET /api/v1/organization/{orgname}/prototypes
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.13.4. createOrganizationPrototypePermission

Create a new permission prototype.

POST /api/v1/organization/{orgname}/prototypes
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Request body schema (application/json)
Description of a new prototype

Name	Description	Schema
role *optional*	Role that should be applied to the delegate	string
activating_user *optional*	Repository creating user to whom the rule should apply	object
delegate *optional*	Information about the user or team to which the rule grants access	object

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14. REPOSITORY

List, create and manage repositories.

2.14.1. listRepos

Fetch the list of repositories visible to the current user under a variety of situations.

GET /api/v1/repository
Authorizations: oauth2_implicit (**repo:read**)

Query parameters

Type	Name	Description	Schema
query	**next_page** *optional*	The page token for the next page	string
query	**repo_kind** *optional*	The kind of repositories to return	string
query	**popularity** *optional*	Whether to include the repository's popularity metric.	boolean
query	**last_modified** *optional*	Whether to include when the repository was last modified.	boolean
query	**public** *optional*	Adds any repositories visible to the user by virtue of being public	boolean
query	**starred** *optional*	Filters the repositories returned to those starred by the user	boolean
query	**namespace** *optional*	Filters the repositories returned to this namespace	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14.2. createRepo

Create a new repository.

POST /api/v1/repository
Authorizations: oauth2_implicit (**repo:create**)

Request body schema (application/json)
Description of a new repository

Name	Description	Schema
repository *optional*	Repository name	string
visibility *optional*	Visibility which the repository will start with	string
namespace *optional*	Namespace in which the repository should be created. If omitted, the username of the caller is used	string
description *optional*	Markdown encoded description for the repository	string
repo_kind *optional*	The kind of repository	

Responses

HTTP Code	Description	Schema
201	Successful creation	

HTTP Code	Description	Schema
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14.3. changeRepoVisibility

Change the visibility of a repository.

POST /api/v1/repository/{repository}/changevisibility
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Change the visibility for the repository.

Name	Description	Schema
visibility *optional*	Visibility which the repository will start with	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14.4. changeRepoState

Change the state of a repository.

PUT /api/v1/repository/{repository}/changestate
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Change the state of the repository.

Name	Description	Schema
state *optional*	Determines whether pushes are allowed.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14.5. deleteRepository

Delete a repository.

DELETE /api/v1/repository/{repository}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14.6. getRepo

Fetch the specified repository.

GET /api/v1/repository/{repository}
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema
query	**includeTags** *optional*	Whether to include repository tags	boolean
query	**includeStats** *optional*	Whether to include action statistics	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.14.7. updateRepo

Update the description in the specified repository.

PUT /api/v1/repository/{repository}
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Fields which can be updated in a repository.

Name	Description	Schema
description *optional*	Markdown encoded description for the repository	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.15. REPOSITORYNOTIFICATION

List, create and manage repository events/notifications.

2.15.1. testRepoNotification

Queues a test notification for this repository.

POST /api/v1/repository/{repository}/notification/{uuid}/test
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**uuid** *required*	The UUID of the notification	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.15.2. deleteRepoNotification

Deletes the specified notification.

DELETE /api/v1/repository/{repository}/notification/{uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**uuid** *required*	The UUID of the notification	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.15.3. getRepoNotification

Get information for the specified notification.

GET /api/v1/repository/{repository}/notification/{uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**uuid** *required*	The UUID of the notification	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.15.4. resetRepositoryNotificationFailures

Resets repository notification to 0 failures.

POST /api/v1/repository/{repository}/notification/{uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**uuid** *required*	The UUID of the notification	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.15.5. listRepoNotifications

List the notifications for the specified repository.

GET /api/v1/repository/{repository}/notification/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.15.6. createRepoNotification

POST /api/v1/repository/{repository}/notification/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Information for creating a notification on a repository

Name	Description	Schema
event *optional*	The event on which the notification will respond	string
method *optional*	The method of notification (such as email or web callback)	string
config *optional*	JSON config information for the specific method of notification	object
eventConfig *optional*	JSON config information for the specific event of notification	object
title *optional*	The human-readable title of the notification	string

Responses

HTTP Code	Description	Schema
201	Successful creation	

HTTP Code	Description	Schema
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.16. REPOTOKEN

Manage repository access tokens (DEPRECATED).

2.16.1. deleteToken

Delete the repository token.

DELETE /api/v1/repository/{repository}/tokens/{code}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**code** *required*	The token code	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.16.2. getTokens

Fetch the specified repository token information.

GET /api/v1/repository/{repository}/tokens/{code}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**code** *required*	The token code	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.16.3. changeToken

Update the permissions for the specified repository token.

PUT /api/v1/repository/{repository}/tokens/{code}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**code** *required*	The token code	string

Request body schema (application/json)
Description of a token permission

Name	Description	Schema
role *optional*	Role to use for the token	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.16.4. listRepoTokens

List the tokens for the specified repository.

GET /api/v1/repository/{repository}/tokens/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.16.5. createToken

Create a new repository token.

POST /api/v1/repository/{repository}/tokens/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Request body schema (application/json)
Description of a new token.

Name	Description	Schema
friendlyName *optional*	Friendly name to help identify the token	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17. ROBOT

Manage user and organization robot accounts.

2.17.1. getUserRobots

List the available robots for the user.

GET /api/v1/user/robots
Authorizations: oauth2_implicit (**user:admin**)

Query parameters

Type	Name	Description	Schema
query	**limit** *optional*	If specified, the number of robots to return.	integer
query	**token** *optional*	If false, the robot's token is not returned.	boolean
query	**permissions** *optional*	Whether to include repositories and teams in which the robots have permission.	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.2. getOrgRobotPermissions

Returns the list of repository permissions for the org's robot.

GET /api/v1/organization/{orgname}/robots/{robot_shortname}/permissions
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.3. regenerateOrgRobotToken

Regenerates the token for an organization robot.

POST /api/v1/organization/{orgname}/robots/{robot_shortname}/regenerate
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.4. getUserRobotPermissions

Returns the list of repository permissions for the user's robot.

GET /api/v1/user/robots/{robot_shortname}/permissions
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.5. regenerateUserRobotToken

Regenerates the token for a user's robot.

POST /api/v1/user/robots/{robot_shortname}/regenerate
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.6. deleteOrgRobot

Delete an existing organization robot.

DELETE /api/v1/organization/{orgname}/robots/{robot_shortname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.7. getOrgRobot

Returns the organization's robot with the specified name.

GET /api/v1/organization/{orgname}/robots/{robot_shortname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.8. createOrgRobot

Create a new robot in the organization.

PUT /api/v1/organization/{orgname}/robots/{robot_shortname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Request body schema (application/json)
Optional data for creating a robot

Name	Description	Schema

Name	Description	Schema
description *optional*	Optional text description for the robot	string
unstructured_metadata *optional*	Optional unstructured metadata for the robot	object

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.9. getOrgRobots

List the organization's robots.

GET /api/v1/organization/{orgname}/robots
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string

Query parameters

Type	Name	Description	Schema
query	**limit** *optional*	If specified, the number of robots to return.	integer
query	**token** *optional*	If false, the robot's token is not returned.	boolean

Type	Name	Description	Schema
query	**permissions** *optional*	Whether to include repostories and teams in which the robots have permission.	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.10. deleteUserRobot

Delete an existing robot.

DELETE /api/v1/user/robots/{robot_shortname}
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.11. getUserRobot

Returns the user's robot with the specified name.

GET /api/v1/user/robots/{robot_shortname}
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.17.12. createUserRobot

Create a new user robot with the specified name.

PUT /api/v1/user/robots/{robot_shortname}
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**robot_shortname** *required*	The short name for the robot, without any user or organization prefix	string

Request body schema (application/json)
Optional data for creating a robot

Name	Description	Schema
description *optional*	Optional text description for the robot	string
unstructured_metadata *optional*	Optional unstructured metadata for the robot	object

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.18. SEARCH

Conduct searches against all registry context.

2.18.1. conductRepoSearch

Get a list of apps and repositories that match the specified query.

GET /api/v1/find/repositories
Authorizations:

Query parameters

Type	Name	Description	Schema
query	**includeUsage** *optional*	Whether to include usage metadata	boolean
query	**page** *optional*	The page.	integer
query	**query** *optional*	The search query.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.18.2. conductSearch

Get a list of entities and resources that match the specified query.

GET /api/v1/find/all
Authorizations: oauth2_implicit (**repo:read**)

Query parameters

Type	Name	Description	Schema
query	**query** *optional*	The search query.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.18.3. getMatchingEntities

Get a list of entities that match the specified prefix.

GET /api/v1/entities/{prefix}
Authorizations:

Path parameters

Type	Name	Description	Schema
path	**prefix** *required*		string

Query parameters

Type	Name	Description	Schema
query	**includeOrgs** *optional*	Whether to include orgs names.	boolean
query	**includeTeams** *optional*	Whether to include team names.	boolean
query	**namespace** *optional*	Namespace to use when querying for org entities.	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.19. SECSCAN

List and manage repository vulnerabilities and other security information.

2.19.1. getRepoManifestSecurity

GET /api/v1/repository/{repository}/manifest/{manifestref}/security
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Type	Name	Description	Schema
path	**manifestref** *required*	The digest of the manifest	string

Query parameters

Type	Name	Description	Schema
query	**vulnerabilities** *optional*	Include vulnerabilities informations	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.19.2. getRepoImageSecurity

Fetches the features and vulnerabilities (if any) for a repository image.

GET /api/v1/repository/{repository}/image/{imageid}/security
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**imageid** *required*	The image ID	string
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema

Type	Name	Description	Schema
query	**vulnerabilities** *optional*	Include vulnerabilities information	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20. SUPERUSER

Superuser API.

2.20.1. listAllUsers

Returns a list of all users in the system.

GET /api/v1/superuser/users/
Authorizations: oauth2_implicit (**super:user**)

Query parameters

Type	Name	Description	Schema
query	**disabled** *optional*	If false, only enabled users will be returned.	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.2. createInstallUser

Creates a new user.

POST /api/v1/superuser/users/
Authorizations: oauth2_implicit (**super:user**)

Request body schema (application/json)
Data for creating a user

Name	Description	Schema
username *optional*	The username of the user being created	string
email *optional*	The email address of the user being created	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.3. listAllLogs

List the usage logs for the current system.

GET /api/v1/superuser/logs
Authorizations: oauth2_implicit (**super:user**)

Query parameters

Type	Name	Description	Schema
query	**next_page** *optional*	The page token for the next page	string
query	**page** *optional*	The page number for the logs	integer
query	**endtime** *optional*	Latest time to which to get logs (%m/%d/%Y %Z)	string
query	**starttime** *optional*	Earliest time from which to get logs (%m/%d/%Y %Z)	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.4. listServiceKeys

GET /api/v1/superuser/keys
Authorizations: oauth2_implicit (**super:user**)

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError

HTTP Code	Description	Schema
404	Not found	ApiError

2.20.5. createServiceKey

POST /api/v1/superuser/keys
Authorizations: oauth2_implicit (**super:user**)

Request body schema (application/json)
Description of creation of a service key

Name	Description	Schema
service *optional*	The service authenticating with this key	string
name *optional*	The friendly name of a service key	string
metadata *optional*	The key/value pairs of this key's metadata	object
notes *optional*	If specified, the extra notes for the key	string
expiration *optional*	The expiration date as a unix timestamp	

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.6. deleteOrganization

Deletes the specified organization.

DELETE /api/v1/superuser/organizations/{name}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**name** *required*	The name of the organizaton being managed	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.7. changeOrganization

Updates information about the specified user.

PUT /api/v1/superuser/organizations/{name}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**name** *required*	The name of the organizaton being managed	string

Request body schema (application/json)
Description of updates for an existing organization

Name	Description	Schema
email *optional*	Organization contact email	string
invoice_email *optional*	Whether the organization desires to receive emails for invoices	boolean

Name	Description	Schema
invoice_email_address *optional*	The email address at which to receive invoices	
tag_expiration_s *optional*	The number of seconds for tag expiration	integer

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.8. approveServiceKey

POST /api/v1/superuser/approvedkeys/{kid}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**kid** *required*	The unique identifier for a service key	string

Request body schema (application/json)
Information for approving service keys

Name	Description	Schema
notes *optional*	Optional approval notes	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.9. deleteServiceKey

DELETE /api/v1/superuser/keys/{kid}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**kid** *required*	The unique identifier for a service key	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.10. getServiceKey

GET /api/v1/superuser/keys/{kid}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**kid** *required*	The unique identifier for a service key	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.11. updateServiceKey

PUT /api/v1/superuser/keys/{kid}
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**kid** *required*	The unique identifier for a service key	string

Request body schema (application/json)
Description of updates for a service key

Name	Description	Schema
name *optional*	The friendly name of a service key	string
metadata *optional*	The key/value pairs of this key's metadata	object
expiration *optional*	The expiration date as a unix timestamp	

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.12. getRepoBuildStatusSuperUser

Return the status for the builds specified by the build uuids.

GET /api/v1/superuser/{build_uuid}/status
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.13. getRepoBuildSuperUser

Returns information about a build.

GET /api/v1/superuser/{build_uuid}/build
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.20.14. getRepoBuildLogsSuperUser

Return the build logs for the build specified by the build uuid.

GET /api/v1/superuser/{build_uuid}/logs
Authorizations: oauth2_implicit (**super:user**)

Path parameters

Type	Name	Description	Schema
path	**build_uuid** *required*	The UUID of the build	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.21. TAG

Manage the tags of a repository.

2.21.1. restoreTag

Restores a repository tag back to a previous image in the repository.

POST /api/v1/repository/{repository}/tag/{tag}/restore
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**tag** *required*	The name of the tag	string

Request body schema (application/json)
Restores a tag to a specific image

Name	Description	Schema
image *optional*	(Deprecated: use **manifest_digest**) Image to which the tag should point	string
manifest_digest *optional*	If specified, the manifest digest that should be used	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.21.2. listTagImages

List the images for the specified repository tag.

GET /api/v1/repository/{repository}/tag/{tag}/images
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**tag** *required*	The name of the tag	string

Query parameters

Type	Name	Description	Schema
query	**owned** *optional*	If specified, only images wholely owned by this tag are returned.	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.21.3. deleteFullTag

Delete the specified repository tag.

DELETE /api/v1/repository/{repository}/tag/{tag}
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Type	Name	Description	Schema
path	**tag** *required*	The name of the tag	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.21.4. changeTag

Change which image a tag points to or create a new tag.

PUT /api/v1/repository/{repository}/tag/{tag}
Authorizations: oauth2_implicit (**repo:write**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**tag** *required*	The name of the tag	string

Request body schema (application/json)
Makes changes to a specific tag

Name	Description	Schema
image *optional*	(Deprecated: Use **manifest_digest**) Image to which the tag should point.	
manifest_digest *optional*	(If specified) The manifest digest to which the tag should point	

Name	Description	Schema
expiration *optional*	(If specified) The expiration for the image	

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.21.5. listRepoTags

GET /api/v1/repository/{repository}/tag/
Authorizations: oauth2_implicit (**repo:read**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Query parameters

Type	Name	Description	Schema
query	**onlyActiveTags** *optional*	Filter to only active tags.	boolean
query	**page** *optional*	Page index for the results. Default 1.	integer
query	**limit** *optional*	Limit to the number of results to return per page. Max 100.	integer
query	**specificTag** *optional*	Filters the tags to the specific tag.	string

Type	Name	Description	Schema

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.22. TEAM

Create, list and manage an organization's teams.

2.22.1. getOrganizationTeamPermissions

Returns the list of repository permissions for the org's team.

GET /api/v1/organization/{orgname}/team/{teamname}/permissions
Authorizations:

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**teamname** *required*	The name of the team	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.22.2. deleteOrganizationTeamMember

Delete a member of a team.

> If the user is merely invited to join the team, then the invite is removed instead.

DELETE /api/v1/organization/{orgname}/team/{teamname}/members/{membername}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**membername** *required*	The username of the team member	string
path	**teamname** *required*	The name of the team	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.22.3. updateOrganizationTeamMember

Adds or invites a member to an existing team.

PUT /api/v1/organization/{orgname}/team/{teamname}/members/{membername}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**membername** *required*	The username of the team member	string
path	**teamname** *required*	The name of the team	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.22.4. getOrganizationTeamMembers

Retrieve the list of members for the specified team.

GET /api/v1/organization/{orgname}/team/{teamname}/members
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**teamname** *required*	The name of the team	string

Query parameters

Type	Name	Description	Schema
query	**includePending** *optional*	Whether to include pending members	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.22.5. deleteOrganizationTeam

Delete the specified team.

DELETE /api/v1/organization/{orgname}/team/{teamname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**teamname** *required*	The name of the team	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError

HTTP Code	Description	Schema
403	Unauthorized access	ApiError
404	Not found	ApiError

2.22.6. updateOrganizationTeam

Update the org-wide permission for the specified team.

PUT /api/v1/organization/{orgname}/team/{teamname}
Authorizations: oauth2_implicit (**org:admin**)

Path parameters

Type	Name	Description	Schema
path	**orgname** *required*	The name of the organization	string
path	**teamname** *required*	The name of the team	string

Request body schema (application/json)
Description of a team

Name	Description	Schema
role *optional*	Org wide permissions that should apply to the team	string
description *optional*	Markdown description for the team	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError

HTTP Code	Description	Schema
404	Not found	ApiError

2.23. TRIGGER

Create, list and manage build triggers.

2.23.1. activateBuildTrigger

Activate the specified build trigger.

POST /api/v1/repository/{repository}/trigger/{trigger_uuid}/activate
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**trigger_uuid** *required*	The UUID of the build trigger	string

Request body schema (application/json)

Name	Description	Schema
config *optional*	Arbitrary json.	object
pull_robot *optional*	The name of the robot that will be used to pull images.	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError

HTTP Code	Description	Schema
404	Not found	ApiError

2.23.2. listTriggerRecentBuilds

List the builds started by the specified trigger.

GET /api/v1/repository/{repository}/trigger/{trigger_uuid}/builds
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**trigger_uuid** *required*	The UUID of the build trigger	string

Query parameters

Type	Name	Description	Schema
query	**limit** *optional*	The maximum number of builds to return	integer

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.23.3. manuallyStartBuildTrigger

Manually start a build from the specified trigger.

POST /api/v1/repository/{repository}/trigger/{trigger_uuid}/start

Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**trigger_uuid** *required*	The UUID of the build trigger	string

Request body schema (application/json)
Optional run parameters for activating the build trigger

Name	Description	Schema
branch_name *optional*	(SCM only) If specified, the name of the branch to build.	string
commit_sha *optional*	(Custom Only) If specified, the ref/SHA1 used to checkout a git repository.	string
refs *optional*	(SCM Only) If specified, the ref to build.	

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.23.4. deleteBuildTrigger

Delete the specified build trigger.

DELETE /api/v1/repository/{repository}/trigger/{trigger_uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**trigger_uuid** *required*	The UUID of the build trigger	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.23.5. getBuildTrigger

Get information for the specified build trigger.

GET /api/v1/repository/{repository}/trigger/{trigger_uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**trigger_uuid** *required*	The UUID of the build trigger	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError

HTTP Code	Description	Schema
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.23.6. updateBuildTrigger

Updates the specified build trigger.

PUT /api/v1/repository/{repository}/trigger/{trigger_uuid}
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string
path	**trigger_uuid** *required*	The UUID of the build trigger	string

Request body schema (application/json)
Options for updating a build trigger

Name	Description	Schema
enabled *optional*	Whether the build trigger is enabled	boolean

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.23.7. listBuildTriggers

List the triggers for the specified repository.

GET /api/v1/repository/{repository}/trigger/
Authorizations: oauth2_implicit (**repo:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.24. USER

Manage the current user.

2.24.1. listStarredRepos

List all starred repositories.

GET /api/v1/user/starred
Authorizations: oauth2_implicit (**user:admin**)

Query parameters

Type	Name	Description	Schema
query	**next_page** *optional*	The page token for the next page	string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.24.2. createStar

Star a repository.

POST /api/v1/user/starred
Authorizations: oauth2_implicit (**repo:read**)

Request body schema (application/json)

Name	Description	Schema
namespace *optional*	Namespace in which the repository belongs	string
repository *optional*	Repository name	string

Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.24.3. getLoggedInUser

Get user information for the authenticated user.

GET /api/v1/user/
Authorizations: oauth2_implicit (**user:read**)

Responses

HTTP Code	Description	Schema
200	Successful invocation	UserView
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.24.4. deleteStar

Removes a star from a repository.

DELETE /api/v1/user/starred/{repository}
Authorizations: oauth2_implicit (**user:admin**)

Path parameters

Type	Name	Description	Schema
path	**repository** *required*	The full path of the repository. e.g. namespace/name	string

Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.24.5. getUserInformation

Get user information for the specified user.

GET /api/v1/users/{username}
Authorizations:

Path parameters

Type	Name	Description	Schema
path	**username** *required*		string

Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	ApiError
401	Session required	ApiError
403	Unauthorized access	ApiError
404	Not found	ApiError

2.25. DEFINITIONS

2.25.1. ApiError

Name	Description	Schema
status *optional*	Status code of the response.	integer
type *optional*	Reference to the type of the error.	string
detail *optional*	Details about the specific instance of the error.	string
title *optional*	Unique error code to identify the type of error.	string
error_message *optional*	Deprecated; alias for detail	string
error_type *optional*	Deprecated; alias for detail	string

2.25.2. UserView

Name	Description	Schema
verified *optional*	Whether the user's email address has been verified	boolean
anonymous *optional*	true if this user data represents a guest user	boolean
email *optional*	The user's email address	string
avatar *optional*	Avatar data representing the user's icon	object
organizations *optional*	Information about the organizations in which the user is a member	array of object
logins *optional*	The list of external login providers against which the user has authenticated	array of object
can_create_repo *optional*	Whether the user has permission to create repositories	boolean
preferred_namespace *optional*	If true, the user's namespace is the preferred namespace to display	boolean

2.25.3. ViewMirrorConfig

Name	Description	Schema
is_enabled *optional*	Used to enable or disable synchronizations.	boolean
external_reference *optional*	Location of the external repository.	string
external_registry_username *optional*	Username used to authenticate with external registry.	
external_registry_password *optional*	Password used to authenticate with external registry.	

Name	Description	Schema
sync_start_date *optional*	Determines the next time this repository is ready for synchronization.	string
sync_interval *optional*	Number of seconds after next_start_date to begin synchronizing.	integer
robot_username *optional*	Username of robot which will be used for image pushes.	string
root_rule *optional*	A list of glob-patterns used to determine which tags should be synchronized.	object
external_registry_config *optional*		object

2.25.4. ApiErrorDescription

Name	Description	Schema
type *optional*	A reference to the error type resource	string
title *optional*	The title of the error. Can be used to uniquely identify the kind of error.	string
description *optional*	A more detailed description of the error that may include help for fixing the issue.	string