# HPE Operations Orchestration

Application Program Interface
(API) Guide

Version 10.60

Document Release Date:  May 2016
Software Release Date: May 2016

**Hewlett Packard Enterprise**

## Legal Notices

### Warranty

The only warranties for HPE products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HPE shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

### Restricted Rights Legend

Confidential computer software. Valid license from HPE required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

### Copyright Notice

### Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation. UNIX® is a registered trademark of The Open Group.

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

## Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to: **http://h20230.www2.HPE.com/selfsolve/manuals**

This site requires that you register for an HPE Passport and sign in. To register for an HPE Passport ID, go to: **http://h20229.www2.HPE.com/passport-registration.html**

Or click the **New users - please register** link on the HPE Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HPE sales representative for details.

## Support

Visit the HPE Software Support Online web site at: https://softwaresupport.HPE.com/
This web site provides contact information and details about the products, services, and support that HPE Software offers.
HPE Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HPE support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

# Table of contents

# Introduction

This document describes HPE Operations Orchestration public Application Programming Interfaces (API).

The public API is HTTP-based. All APIs are RESTful and use JavaScript Object Notation (JSON).

## REST API Versions

Some previous releases of HPE OO 10 have introduced new REST API versions. When a new version is introduced, previous versions are deprecated, but still supported for a period of time. The current version is v2 and the APIs documented in this guide are updated for it. The version should be specified as a prefix to all REST calls (e.g. /oo/rest/v2/executions).

**Note**: New HPE OO releases may introduce new optional fields to returned representations that are not considered as an API break.

Therefore, the client should be tolerant to new attributes when deserializing JSONs. For example, when performing a GET operation on /test, the following is returned:

```
{
"msg" : "hello world"
}
```

We may add a new attribute in the next version and then the GET on /test will return the following, in this case the client should not break:

```
{
"msg" : "hello world",
"msg2" : "It's a beautiful day! "
}
```

## Deprecated API's

HPE is committed to support deprecated APIs until the next major release.

## Basic Concepts

See the Concepts Guide for more information on the basic concepts of HPE Operations Orchestration.

## RESTful APIs

All REST APIs have a prefix of `/oo/rest/v2/`. For example, `POST /oo/rest/v2/executions.`

### Request Headers

The `content-type` and `accept` headers are usually added for every request.

The `content-type` represents the MIME (RFC2045) type of the request body. The `content-type` is usually `application/json` unless otherwise stated in a specific API.

The `accept` header represents the requested format of the response from the Central server. The `accept` header is also usually `application/json` unless mentioned differently.

## Integration Use Case

This chapter describes a common usage of the HPE OO API and comes to demonstrate its capabilities. Keep in mind that use case described here is only one example on a common use case of HPE OO platform integration. HPE OO APIs allow much more than that.

### Use Case Description

The most common use case when integrating with HPE OO is allowing various types of end users to invoke automation using organizational portal or a third party application. For example, to remediate an incident, doing routine tasks like reset password for a user or creating a DB schema in Dev environment, and so on.

The following implementation is a suggestion and can be adopted at any level you see fit.

### Use Case Implementation

The integration includes two separate processes. These processes are described from the user perspective, but also describe the work to be done by the integration developer.

## Portal/App Admin Process - Selecting Flows to Expose to Users

**Process description**

Before the user of the Organizational Portal/Application will be able invoke flows from it, the Admin needs to determine which flows he would like to expose to the user and for each one of them to determine from where the user is able to invoke and assign data sources for the flow inputs.

The Admin experience is:
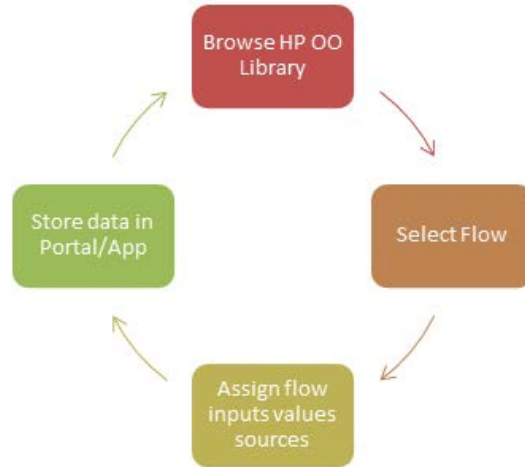


## Process implementation

This process, if used as described, requires UI development on the portal/application side in order to allow the Admin to browse the library and select a flow.

For example:

• Drop down selection box that lists all the flows in a specific folder in the library (means that the path will need to be decided in advance).

• A folders tree graphical window like the following:

Another option which is less usable for the Admin is only supplying the UI that allows the Admin to manually insert the flow UUID and input parameters value sources.

The following table describes how the implementation of the interactions with HPE OO Central server look like.

**Step**: Browse HPE OO Library

**Admin Action**: Browse the content library from the portal/application.

Integrator Actions (interaction with OO): Lists the folders and flows under a given path while the root of the path is the HPE OO Content Library, which is 'Library/'. In this example, if the organizational portal team decides to implement in the UI the full library tree display (as shown in the image above), the integrator code is required to be recursive. That is, a REST call will be implemented for every branch that the end user clicks.

Looking at the example in the image above, the first REST call was to list the top level libraries, then when the user clicked on 'Accelerator Packs' a REST call was submitted to list the levels below it, and so on.

See API: `GET /flows/tree/level`

**Step**: Select Flow

**Admin Action**: Select flows to invoke in order to make them available in the portal/application and also define where.

**Integrator Actions** (interaction with OO): Get the selected flow details like UUID, Inputs, Description, etc. The details that will be collected depend on what information was decided to display to the Admin in the UI. For invoking the information needed is UUID and inputs information.

See API: `GET /flows/{uuid}`

**Step**: Assign flow inputs values sources

**Admin Action**: Bind value sources to the flow inputs. The sources will most likely be dynamic objects from the application data (like internal variable, called SelectedItemHostname) and not static values.

Integrator Actions (interaction with OO): Provide the capability for this in the portal/application. Note: A validation will need to be implemented to make sure the Admin will provide value source to each of the flow inputs that are marked as Prompt User. Otherwise the flow will pause and will wait for inputs, for example, OO Admin will need to login to Central and enter them.

See API: `GET /flows/{uuid}/inputs`

**Step**: Store data in Portal/App

**Admin Action**: Store all the information in the Portal/Application.

**Integrator Actions** (interaction with OO): Save the relevant data to the portal/application (in its DB/Forms/Files/etc.)

**Note**: The flow UUID, inputs and their value source must be kept on the Portal/Application side for the flow invocation.

### End User process – Invoking and Monitoring Workflows

**Process description**

This process occurs in the organizational portal or the third party applications, on the area that is exposed to the end user. The best practice is to have one place that holds the functionality, like an internal service, so the other areas of the application that allow users to trigger flows calls it over and over.
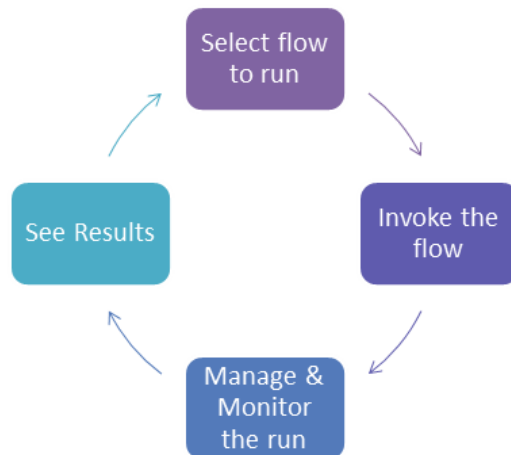


## Process implementation

The following table describes how the implementation of the interactions with HPE OO Central server looks like.

The interaction is done through the HPE OO REST API.

See Flow Execution for more details.

**Step**: Select flow to run.

**End User Action**: From the portal/application, the end user will select the flow to invoke from a predefined list or just click on a button that the admin made available.

Integrator Actions (interaction with OO): Collect the information to be used later for invoking the flow. This includes UUID of the flow selected and input parameters designated values.

**API to use**: None

**Step**: Invoke the flow.

**End User Action**: The workflow will be invoked while the portal/application will feed it with the needed input values.

**Integrator Actions** (interaction with OO): Use the REST API to invoke the flow. Use the UUID and the flow input parameters names and values. It is also recommended to use the `runName` invocation parameters in order to allow better troubleshooting later on. A suggested format for the `runName` can be:

`<InvokingAppName>:<InvokingUserName>:<TargetSystemName>:<ActionName>`

**API to use**: Execute a Flow by UUID `POST /executions`

**Step**: Manage and Monitor the run.

**Integrator Actions** (interaction with OO): Uses the following options:

1. A code that includes a loop that continuously calls HPE OO to get the status summary.

2. For being able to display to the end user progress of the run in addition to the status, develop code that performs as stated above and in addition, for the progress, continuously requests the current running steps.

**API to use**:

1. Get a single execution's summary:

   – Use GET /executions/{executionsId}/summary.
   – You can also use GET /executions/{executionId}/execution-log to have the above with additional information on inputs and outputs.

2. Get status on set of executions:

   – To get a short run summary on a list of executions with filtering capabilities such as – status, run name, flow UUID, use the Get Executions API: GET /Executions. This can be used for example for Grid view.

3. Get Execution Steps (with Pagination and Filtering):

   – To see run's execution steps, use the Get Execution Step (with Pagination and Filtering) API:
     • GET /executions/{executionId}/steps.
     • This allows you to see all run's executed steps, get their inputs, outputs and more details.

This information allows you to track current running execution and also use this to debug your run.

**End User Action**: Control the run (Optional). Users can take the following actions on the run:

1. Pause the run.

2. Resume the run.

3. Cancel the run.

Integrator Actions (interaction with OO): Implementing some or all of this will provide more control to the end user, which can be very helpful to some end user types. But on the other hand need to have additional UI development on the portal/application side. When implementing Pause and Resume make sure to keep track on the run status after Resume action was activated.

**API to use**:

• Change the Status of an Execution

## Basic Authentication

When user authentication is on, the client must provide their credentials when calling the REST APIs. Central supports preemptive basic authentication.

The client should add a header with the following key/value:

• Key: Authorization

• Value: Basic base64 (username:password)

For example, the authorization value for `admin:1234` is:

`Basic YWRtaW46MTIzNA==`

On an unsuccessful authentication attempt, the service returns an HTTP 401 code.

## Permissions

If the user does not have the assigned permission to activate an API the following status code appears:

`403 – Forbidden. The user attempting to execute this command does not have the needed permission.`

**Note**: This applies to all the API's in the version.

# CSRF Protection in HPE OO 10.x

A CSRF (Cross-Site Request Forgery) attack is when a malicious web site, email, blog, instant message, or program causes a user's web browser to perform an unwanted action on a trusted site for which the user is currently authenticated. The impact of a successful cross-site request forgery attack is limited to the capabilities exposed by the vulnerable application.

In a fresh installation of HPE OO, or when you upgrade from an earlier version, CSRF protection is enabled by default.

For more information about CSRF protection, see https://www.owasp.org/index.pHPE/Cross-Site_Request_Forgery_%28CSRF%29_Prevention_Cheat_Sheet#Disclosure_of_Token_in_URL .

**Important**: If you are working with a REST client that retains the session, continue reading this section. If not, then this section is not relevant for you.

## Adding the CSRF Token Header

When using REST APIs, in some situations, you will need to add the CSRF token header (just for POST/DELETE/PUT).

The **POST**, **PUT**, and **DELETE** requests are protected.

If you are working with a REST client that keeps the session you will need to provide the CSRF token after the first call.

Following is an example for a Google Chrome REST console:

1. **First** POST works (for example, create system configuration):



2. After **second** POST:



3. Now provide the CSRF token on the request. Get the **X-CSRF-TOKEN** from the server response. You can use a simple dummy **GET** to give you the token value in the response cookies:



4. Now you can **POST** with this token in the header:

Following are the results:



**Example of handling CSRF token in code**

In Java, get the X-CSRF-TOKEN from the response **cookies**:

**Example for org.apache.http.client**

```java
private void handleCsrfToken(BasicCookieStore cookieStore, HttpRequestBase request) {

    // handle CSRF token

    if (cookieStore != null) {

        String token = null;

        for (Cookie cookie : cookieStore.getCookies()) {

            if (cookie.getName().equals("X-CSRF-TOKEN-OO")) {

                token = cookie.getValue();

                break;

            }

        }


        if (token != null) {

            request.setHeader("X-CSRF-TOKEN", token);

        }

    }

}
```

In Java, get the X-CSRF-TOKEN from the response headers:

**Example for org.apache.http.client**

```java
private void handleCsrfToken(HttpRequestBase requestBase) throws Exception {

    HttpGet httpGet = new HttpGet(connectCommand.getUrl() + "/rest/latest/users/me");

    HttpRequestResult result = doSimpleRequest(httpGet); // dummy get

    Header[] headers = result.getResponse().getHeaders("X-CSRF-TOKEN");


    if (headers != null && headers.length > 0) {

        requestBase.addHeader(headers[0].getName(), headers[0].getValue());
```

```
        }

}
```

JavaScript example:

```
var header = xhr.getResponseHeader('X-CSRF-HEADER'); // get CSRF header name from response


var token = xhr.getResponseHeader('X-CSRF-TOKEN'); // get CSRF token value from response

xhr.setRequestHeader(header || 'X-CSRF-TOKEN', token || ''); // set CSRF header in the
request
```

If you need to temporarily disable CSRF protection in order to fix your API calls, perform the following. It is recommended that you enable the CSRF protection once the required changes have been implemented.

You can enable or disable CSRF protection by setting the `csrf.protection.enabled` system configuration item to true or false.

This can be done via REST API. For more information, see "Update System Configuration Item" in the *HPE OO API Guide*.

To set this system configuration item, you must have Manage System Settings permissions.

You can also enable or disable CSRF protection using the HPE OO Shell (OOSH) tool. Use the following command:

```
oosh> set-sys-config –key csrf.protection.enabled –value true ……
```

Make sure to specify the options (`--key, --value, --url, --user, --password`) for this command.

For more information, see the *HPE OO Shell (OOSH) User Guide*.

## Backward Compatibility with HPE OO 9.x APIs

Some SOAP and REST APIs from HPE OO 9.x are supported by HPE OO 10.x, and some are not.

Some of the APIs from HPE OO 9.x have equivalent REST APIs for HPE OO 10.x. We recommend using the REST APIs for HPE OO 10.x.

- The base path for using HPE OO 10.x REST API is:
  http(s)://<OO Central Server Name / IP>:<PORT>/oo/rest/.
- The URL for using HPE OO 9.x SOAP API while working with OO 10.x Central is the same as in HPE OO 9.x:
  https:// <OO Central Server Name / IP>:<PORT>/PAS/services/WSCentralService.
- The URL for using HPE OO 9.x REST API while working with OO 10.x Central is the same as in HPE OO 9.x:
  https:// <OO Central Server Name / IP>:<PORT>/PAS/services/rest.

## SOAP

In the following table you can find information on what is supported, what is not, and the HPE OO 10.x API that we recommend to use. For details on the HPE OO 10.x REST requests, see the section below.

| Functionality | 9.x Request | 10.x Support for 9.x Request | 10.x Equivalent REST Request |
|---|---|---|---|
| **Configurations** | getLWSSOConfig | Not Supported | GET /authns /lwsso-config |
| | updateLWSSOConfig | Not Supported | PUT /authns /lwsso-config |
| **Clusters** | getClusterNodes | Not Supported | N/A |
| **Flows** | getFlowDetails | Supported | GET /flows/{uuid} |
| | getFlowGraph | Partially supported. The request will succeed, but a static image is returned saying that this feature is not supported. | N/A |
| | getFlowInputDescriptions | Not Supported | GET /flows/{uuid}/inputs |
| **Groups and User Management** **Note**: In HPE OO 10.x, user groups are called user roles. | createGroup | Not Supported | POST /roles |
| | updateGroup | Not Supported | PUT /roles/{roleName} |
| | deleteGroup | Not Supported | DELETE /roles/{roleName} |
| | getUserGroups | Not Supported | GET /roles |
| | createUser | Not Supported | POST /users |
| | updateUser | Not Supported | PUT /users/{username} |
| | deleteUser | Not Supported | DELETE /users/{userIds} |
| **Repositories** **Note**: In HPE OO 10.x, the concept of repository was replaced with new concepts. See the HPE OO Concepts Guide. | getPermissions | Not Supported | In order to control content permissions, use: GET /roles/{rolesNames}/entitlements/{path} or PUT /roles/{roleName}/entitlements/{path}. In order to control content configuration items review the /config-items/ API section. Note you cannot change the content items themselves from the API in OO 10.x. |
| | setPermissions | Not Supported | |
| | getAttributes | Not Supported | |
| | renameRepoEntity | Not Supported | |
| | deleteRepoEntity | Not Supported | |
| | moveFlow | Not Supported | |
| | updateDescription | Not Supported | |
| | createFolder | Not Supported | |
| | moveFolder | Not Supported | |
| | list | Supported | Get Flows Library GET /flows/tree/level |
| | search | Supported | N/A |

| Functionality | 9.x Request | 10.x Support for 9.x Request | 10.x Equivalent REST Request |
|---|---|---|---|
| **Runs** | getFlowsRunHistory | Not Supported | N/A |
| | getFlowRunHistory | Supported | GET /executions |
| | pauserun | Supported | PUT /executions/{executionId}/status |
| | resumerun | Supported | PUT /executions/{executionId}/status |
| | cancelrun | Supported | PUT /executions/{executionId}/status |
| | runFlow | Supported | POST /executions |
| | runFlowEx | Supported | POST /executions |
| | getRunStatus | Supported | GET /executions/{executionIds}/summary |
| | getRunStatusEx | Supported | GET /executions/{executionIds}/summary |
| | getStatusForRuns | Not Supported | N/A |
| **Scheduler** | isScheduledFlowPaused | Not Supported | GET /schedules/ <br> GET /schedules/{id} |
| | isSchedulerPaused | Not Supported | GET /schedules/ <br> GET /schedules/{id} |
| | isSchedulerEnabled | Not Supported | GET /schedules/ <br> GET /schedules/{id} |
| | getSchedulesForFlowCategory | Not Supported | N/A |
| | pauseScheduledFlow | Not Supported | PUT /schedules/{ids}/enabled |
| | pauseSchedule | Not Supported | PUT /schedules/{ids}/enabled |
| | resumeSchedule | Not Supported | PUT /schedules/{ids}/enabled |
| | scheduleFlow | Not Supported | POST /schedules |
| | getSchedule | Not Supported | GET /schedules/{id} |
| | deleteSchedule | Not Supported | DELETE /schedules/{ids} |
| | getScheduledFlows | Not Supported | GET /schedules/ |
| | getSchedulesOfFlow | Not Supported | GET /schedules/ |
| | resumeScheduledFlow | Not Supported | PUT /schedules/{ids}/enabled |
| | deleteScheduledFlow | Not Supported | GET /schedules/ <br> DELETE /schedules/{ids} |
| **Selection Lists** | getSelectionList | Not Supported | N/A |
| | createSelectionList | Not Supported | N/A |
| **Repositories** | /list/{path} | Supported | N/A |
| **Runs** | /run/{flow path/uuid} | Supported | POST /executions |

## REST

| Functionality | 9.x Request | 10.x Support for 9.x Request | 10.x Equivalent REST Request |
|---|---|---|---|
| Repositories | /list/{path} | Supported | Get Flows Library<br>GET /flows/tree/level |
| Runs | /run/{flow path/uuid} | Supported | POST /executions |

## External Links

- Report:

  ```
  https://localhost:8451/PAS/app?service=RCLinkService/ReportLinkDispatch&sp=SINDIVIDUAL_R
  EPAIR_LEVEL&sp=S${flowUUID}&sp=l0&sp=l${historyId}&sp=l${runId}
  ```

  ```
  https://localhost:8451/PAS/app?service=RCLinkService/FlowLinkDispatch&sp=SRESUME&sp=l${r
  unId}
  ```

- Guided Run:

  ```
  https://localhost:8451/PAS/app?service=RCLinkService/FlowLinkDispatch&sp=SNEWRUN&sp=S${f
  lowUUID}&sp=l0&sp=S${flowName}_${runID}
  ```

- Run All:

  ```
  https://localhost:8451/PAS/app?service=RCLinkService/FlowLinkDispatch&sp=SNEWRUNALL&sp=S
  ${flowUUID}&sp=l0&sp=S${flowName}_${runID}
  ```

  ${flowName}_${runID} - can be empty.

- Instant Run:

  ```
  https://localhost:8451/PAS/app?service=RCLinkService/InstantRun&sp=S${flowUUID}&sp=l0&sp
  =S${flowName}_${runID}
  ```

  ${flowName}_${runID} - can be empty.

## Changes in HPE OO version 10.60

### Added

| API | Description |
|-----|-------------|
| GET /ras-upgrade-file/version | Returns the version of the RAS upgrade file that has been uploaded to Central |
| POST /ras-upgrade-file | Uploads the RAS upgrade file to Central. The uploaded file can later be used to upgrade RASes via Central. The file will be overwritten if it already exists. |
| DELETE /ras-upgrade-file | Deletes the RAS upgrade file. |
| GET /rases | Retrieves the information of all the RASes in the system. |
| POST /rases/{uuids}/upgrade | Initiates an upgrade action (either prepare upgrade or start upgrade) for the specified RASes. |
| POST /reverse-rases | Connect to and register reverse RAS. |
| PUT /reverse-rases/{uuid} | Update reverse RAS details. |
| POST /reverse-rases/test | Open connection to the worker and close it after success |
| GET /reverse-ras-proxy | Retrieve proxy configuration for reverse RAS connection. |
| POST /reverse-ras-proxy | Create a reverse RAS proxy configuration. |
| PUT /reverse-ras-proxy | Update reverse RAS proxy configuration. |
| DELETE /reverse-ras-proxy | Delete proxy configuration for reverse RAS connection. |

### Changed

| API | Changes |
|-----|---------|
| GET /executions/{executionId}/steps | pageSize is limited to 1000 instead of 10000 |

# REST APIs

This section includes the RESTful APIs used in HPE Operations Orchestration.

# Flow Execution

These APIs enable you to execute flows.

## Get Executions Summary (with pagination and filtering)

**Request**: `GET /executions`

**Description**: Returns a paginated list of executions summary, with filtering. The returned objects include the execution summary, detailed objects with data about the execution.

**Request parameters**:

| Attribute | Type | Description | Required |
|---|---|---|---|
| pageNum | Integer | The number of the returned page.<br> Default value: 1 | No |
| pageSize | Integer | The size of the returned page.<br>Default value: 200 | No |
| flowPath | String | Return runs that their flow path contains this string. | No |
| status | Array of Predefined Values | Return runs with the following statues.<br>Possible values:<br>`RUNNING, COMPLETED, COMPLETED_ RESOLVED, COMPLETED_DIAGNOSED, COMPLETED_ERROR, COMPLETED_NO_ACTION_TAKEN, COMPLETED_CUSTOM, SYSTEM_FAILURE, PAUSED, PAUSED_USER_PAUSED, PAUSED_INPUT_REQUIRED, PAUSED_ INPUT_REQUIRED_MANUAL_OP, PAUSED_DISPLAY, PAUSED_GATED_ TRANSITION, PAUSED_HAND_OFF, PAUSED_INTERRUPT, PAUSED_NO_ WORKERS_IN_GROUP, PAUSED_ BRANCH_PAUSED, CANCELED` | No |
| owner | String | Return runs when the owner name contains this string. | No |
| runName | String | Return runs with a run name that contains this string. | No |
| runId | String | Return runs when the run id contains this string. | No |
| flowUuid | String | Return runs where the flow UUID contains this string. | No |
| startedAfter | Long | Returns runs where the start time is after this time. | No |
| startedBefore | Long | Returns runs where the start time is before this time. | No |

**Examples:**

```
GET /executions?pageNum=1&pageSize=100&flowUuid=4c&flowPath=Library/cp-
filters&status=COMPLETED_RESOLVED&owner=anonymous&runName=getAllFilter&runId=30
```

```
GET /executions?pageNum=4&pageSize=15&status=PAUSED_INPUT_REQUIRED,PAUSED_HAND_OFF
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | Returns empty list if nothing found or not viewable. |
| 400 | Bad Request | Illegal arguments. One of the arguments is not valid. |

**Response entity body:**

An array which contains all of the executions summary which match the requested filters. Each element in the array represents an execution.

The returned array is ordered by descending starting times. Therefore, the execution that has the latest starting time will be first, and would be empty if no execution exists for the given filters.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| executionId | String | The run ID. | |
| branchId | String | The branch id (if it exists). | |
| startTime | Date | The run start time. | |
| endTime | Date | The run end time ( if ended) | |
| status | Predefined Value | The status of the execution. Possible values:<br>RUNNING, COMPLETED, SYSTEM_ FAILURE, PAUSED, CANCELED<br><br>See Flow Execution Status | |
| resultStatusType | String | The run result (if ended). | |
| resultStatusName | String | The run result (if ended). | Is null if status is not "COMPLETED" |
| pauseReason | Predefined Value | The reason for the pause. Possible values:<br>USER_PAUSED, INPUT_ REQUIRED, INPUT_REQUIRED_ MANUAL_OP, DISPLAY, GATED_ TRANSITION, HAND_OFF, INTERRUPT, NO_WORKERS_IN_ GROUP, BRANCH_PAUSED | Is null if status is not "PAUSED" |
| owner | String | The run owner. | |
| triggeredBy | String | The domain\user that triggered the execution. If the domain is an LDAP domain that was deleted from OO, the domain will be shown as N/A. | |
| flowUuid | String | The uuid of the flow which was executed. | |
| flowPath | String | The path of the flow which was executed. | |
| executionName | String | The name of the execution. | |

| roi | Double | The value of the ROI of the execution. | |
|---|---|---|---|
| ownerDomain | String | The name of the domain on which the run owner belongs to. If the domain is an LDAP domain that was deleted from OO, the ownerDomain will be N/A. | |
| triggeringSource | String | The source of which the run was triggered (scheduler, studio, and do on). | |

Example:

```
          "executionId":"100749",
          "branchId":null,
          "startTime":1371106274153,
          "endTime":1371106277160,
          "status":"COMPLETED",
          "resultStatusType":"RESOLVED",
          "resultStatusName":"success",
          "pauseReason":null,
          "owner":"anonymousUser",
          "ownerDomain:null"
          "triggeredBy":"anonymousUser",
          "flowUuid":"06fe8531-868b-4e79-aa7a-13a5e30a66ec",
          "flowPath":"Library/Utility Operations/Samples/Generate/Number.xml",
          "executionName":"Generate Random Number",
          "triggeringSource":"central"
          "roi":null
      },
      {
          "executionId":"100267",
          "branchId":null,
          "startTime":1371104522563,
          "endTime":1371104576253,
          "status":"COMPLETED",
          "resultStatusType":"ERROR",
          "resultStatusName":"failure"
          , "pauseReason":null,
          "owner":"anonymousUser",
          "ownerDomain:null"
          "triggeredBy":"anonymousUser
          ",
          "flowUuid":"1901edde-3cac-4da6-915c-fd254e23169c",
          "flowPath":"Library/Multihost Connectivity Diagnostic.xml",
          "executionName":"Multihost Connectivity Diagnostic",
          "triggeringSource":"central"
          "roi":null
      }
  ]
```

# Get Execution Summary

**Request**: `GET /executions/{executionIds}/summary`

**Description**: Retrieves the details of a specific execution.

**Example**:

`GET /executions/3332190961082830376,679861347442169334/summary`

**Request path variables:**

| Attribute | Description |
|---|---|
| executionIds | The ids of the executions |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | The requested execution log was. |
| 403 | Forbidden | |
| 404 | Not Found | The requested execution log was not found. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
[{
        "executionId":"3332190961082830376",
        "branchId":null,
        "startTime":1371475041169,
        "endTime":null,
        "status":"PAUSED",
        "resultStatusType":"RESOLVED",
        "resultStatusName":"HAHA",
        "pauseReason":"USER_PAUSED",
        "owner":"anonymous",
        "ownerDomain:null",
        "triggeredBy":"anonymous",
        "flowUuid":"a8e8fc10-b584-4d39-921f-987b29c9dd19", "flowPath":null,
        "executionName":"mock flow",
        "triggeringSource:central" "roi":null
},
{
        "executionId":"679861347442169334",
        "branchId":null,
        "startTime":1371475041169,
        "endTime":null,
        "status":"PAUSED",
        "resultStatusType":"RESOLVED",
        "resultStatusName":"HAHA",
        "pauseReason":"USER_PAUSED",
        "owner":"anonymous",
        "ownerDomain:null",
        "triggeredBy":"anonymous",
        "flowUuid":"a8e8fc10-b584-4d39-921f-987b29c9dd19",
        "flowPath":null,
        "executionName":"mock flow",
        "triggeringSource:central"
        "roi":null
}
]
```

See returned items in the Get Execution API for more information.

## Get Execution Step Count

**Request**: `GET /executions/{executionId}/steps/count`

**Description**: Returns the total number of executed steps for the given execution, including finished steps and currently executing (or paused) steps. If a step has been executed more than once (e.g., in a loop), this will be reflected in the result. The count includes virtual steps such as lanes.

This command is useful when retrieving steps with pagination. It allows one to compute the total number of pages, or the page number in which a given step could be found.

**Request path variables:**

| Attribute | Description |
|---|---|
| `executionId` | Execution ID whose steps to count. |

**Request parameters:**

| Attribute | Type | Description | Required | Default |
|---|---|---|---|---|
| `upToPath` | String | If provided, the result will only count steps positioned before this path (exclusive). The given value is a step path in the execution tree: The paths of the steps at the top level of the flow are 0.0 for the 1st step, 0.1 for the 2nd and so on. If the 2nd step (0.1) is a subflow, then the paths of the steps in that subflow are 0.1.0, 0.1.1 and so on. | No | N/A |

**Example:**

`GET /executions/100300001/steps/count?upToPath=0.7.0.13`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | Successful. If there are no results, the result will be empty, but this is still OK. |
| 400 | Bad Request | If any of the arguments are invalid. |
| 404 | Not Found | If the execution cannot be viewed or does not exist. |

# Get Execution Steps (with Pagination and Filtering)

**Request**: `GET /executions/{executionId}/steps`

**Description**: Returns a paginated list of executed steps, with optional filtering. If a step has been executed more than once, for example, loop, this will be reflected in the result. The result includes virtual steps such as lanes.

The returned objects are step logs – detailed objects with all available data about the steps. Step logs are created as soon as the step begins to execute, and are updated with more data when the step finishes.

**Request path variables:**

| Attribute | Description |
|---|---|
| `executionId` | Execution ID whose steps to retrieve. |

**Request parameters – pagination (optional):**

| Attribute | Type | Description | Default | Required |
|---|---|---|---|---|
| `pageNum` | Long | The wanted page number (1 or greater). | 1 | No |
| `pageSize` | Long | The size of each page – can be 1 to 1000. | 50 | No |
| `order` | Predefined Value | The order of the returned steps – asc for ascending or desc for descending. The order is by step path. | asc | No |

**Request parameters – filtering criteria:**

General notes about filtering:

- All filtering parameters are optional
- Only steps that satisfy all of the given criteria will be returned
- Filtering is performed before the pagination. In other words, when filtering, the pagination will give you pages of search results.
- Filtering by text is case insensitive
- Range conditions (such as roiFrom and roiUpTo) are exclusive – see the example below for details.

| Attribute | Type | Description |
|---|---|---|
| `path` | String | Get the step with this exact path. See Get Execution Step Count. |
| `pathFrom` | String | Get steps whose paths are greater than this path. Greater refers to a collapsible tree representation of the execution, the greater path would show up lower in the tree. For example, 0.10.0 is greater than both 0.10 (its parent) and0.9.99999. |
| `pathUpTo` | String | Get steps whose path is less than this path (also see `pathFrom`). |
| `nameContains` | String | Get steps whose names contain this substring. |
| `types` | Predefined Value | Get steps of the given types (comma-separated list, no spaces). The types are: `operation, subflow, return_step, other.` `other` represents special steps like Multi Instance, as well as lanes. |
| `startTime` | Long | Get steps that started at this exact timestamp. Timestamps are "Unix time" numbers with millisecond resolution (the number of milliseconds elapsed since 00:00:00 1 January 1970 UTC). |
| `startTimeFrom` | Long | Get steps that started after this timestamp. |

| Attribute | Type | Description |
|---|---|---|
| startTimeUpTo | Long | Get steps that started before this timestamp. |
| endTime | Long | Get steps that ended at this exact timestamp. |
| endTimeFrom | Long | Get steps that ended after this timestamp. |
| endTimeUpTo | Long | Get steps that ended before this timestamp. |
| durationSec | Long | Gets steps with this exact execution time (seconds). |
| durationSecFrom | Long | Get steps that took longer to execute than this value (seconds). |
| durationSecUpTo | Long | Get steps that took less time to execute than this value (seconds). |
| inputsContain | String | Get steps where any of the inputs contain this sub-string, in either the input name or its value. It is also possible to make a search in the form `name=value`.<br><br>**Note**: Values over 4,000 bytes cannot be searched. |
| rawResultsContain | String | Get steps where any of the raw results contain this sub- string, in either the result name or its value. It is also possible to make a search in the form `name=value`.<br>**Note**: Values over 4,000 bytes cannot be searched. |
| primaryResultContains | String | Get steps where the primary result contains this sub- string.<br>**Note**: Values over 4,000 bytes cannot be searched. |
| stepResultsContain | String | Get steps where any of the step-defined results contain this sub-string, in either the result name or its value. It is also possible to make a search in the form name=value.<br>**Note**: Values over 4,000 bytes cannot be searched. |
| responseTypes | Predefined Value | Get steps having one of the given response types (comma-separated list, no spaces).<br>The types are: `resolved, error, diagnosed, no_action_taken, exception`.<br>`exception` means the step's execution could not be completed. |
| transitionContains | String | Get steps whose outgoing transition messages contain this sub-string. The transition message is either the transition's description, or the transition's name if no description has been defined.<br>**Note**: Transition descriptions over 4,000 bytes (for all locales combined) cannot be searched. |
| roi | Double | Get steps whose outgoing transitions have this exact ROI value. |
| roiFrom | Double | Get steps whose outgoing transitions have ROI values greater than this. |
| roiUpTo | Double | Get steps whose outgoing transitions have ROI values less than this. |
| currentFlowContains | String | Get steps that belong to flows whose names contain this sub-string. |
| currentFlowIdContains | String | Get steps that belong to the flow with the given ID (or a sub-string in it). |
| stepIdContains | String | Get steps with the given static step ID (or a sub-string in it). This ID is hard-coded in the parent flow definition. **Note** that the same step ID may appear in multiple steps of the execution, for example, when the flow employs a loop. To get a specific step in the execution tree, use the **path** parameter. |

| Attribute | Type | Description |
|---|---|---|
| invokedIdsContain | String | Get steps that invoke (execute) the operation or subflow with the given ID (or a sub-string in it).<br>Note: for steps that run soft copies of operations, the invoked IDs include both the soft copy and its parent operation. |
| userContains | String | Get steps that executed while the given Central user name (or a sub-string in it) was the execution owner. |
| workerIdContains | String | Get operation-type steps that were executed by the given worker UUID (or a sub-string in it). |
| workerGroupContains | String | Get operation-type steps that were executed by a worker in the given worker group name (or a sub-string in it). |

Example:

```
GET /executions/100300001/steps?pageNum=3&pageSize=20&order=desc
        &pathFrom=0.7.2&pathUpTo=0.9&path=0.9
        &nameContains=ping
        &types=operation,subflow
        &startTime=1391359774000&startTimeFrom=1391359774000&startTimeUpTo=13913597800
        00
        &durationSecFrom=300&durationSecUpTo=400&durationSec=400
        &inputsContain=localhost
        &resultsContain=ping+completed
        &responseTypes=resolved,diagnosed
        &transitionContains=success
        &roi=7.5&roiFrom=7.5&roiUpTo=30
        &currentFlowContains=My+Subflow
        &userContains=jon
        &workerIdContains=5c2002da
```

This will search execution 100300001 for steps satisfying all of the criteria, and return the 3rd page out of the result set (with 20 steps per result page). The steps will be searched in descending path order, starting with the step that has the greatest path (normally the main flow's return step).

The combination of these three conditions: `roi=7.5&roiFrom=7.5&roiUpTo=30`, will search for: 7.5 ≤roi < 30.

Following are the possible search ranges:

| Use paramaters… | To search for… |
|---|---|
| roi=7.5 | roi = 7.5 |
| roiFrom=7.5 | roi > 7.5 |
| roi=7.5&roiFrom=7.5 | roi ≥ 7.5 |
| roiFrom=7.5&roiUpTo=30 | 7.5 < roi < 30 |
| roi=7.5&roiFrom=7.5&roiUpTo=30 | 7.5 ≤ roi < 30 |
| roiUpTo=7.5&roiFrom=30 | roi < 7.5 or roi > 30 |
| roiUpTo=7.5&roiFrom=7.5 | roi ≠ 7.5 |

**Response Attributes:**

**StepInfo object:**

This object is one of the attributes of a step. It represents a summary of the main step details.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| stepId | String | Step UUID | |
| stepName | String | Step name | |
| path | String | Step path within the execution tree. | See Get Execution Step Count for a description of step paths |
| responseType | Predefined Value | Same as in the StepTransitionLog object. | |
| startTime | Long | Step execution start time. | |
| endTime | Long | Step execution end time. | |
| paused | Boolean | true if this step's execution is currently paused, false otherwise. | |
| orderNumber | String | Sortable hexadecimal representation of path. | |
| invokedIds | Array of strings | The UUID of the operation or subflow invoked by this step (if any). | If the step invoked a soft copy of an operation, both the copy and the parent operation are included. |
| flowName | String | Name of the flow containing this step. | |
| flowId | String | UUID of the flow containing this step. | |
| type | Predefined Value | Step type. The list of possible types depends on the OO version. The current types are: OPERATION, SUBFLOW, RETURN_STEP, PARALLEL, MULTI_INSTANCE, NON_BLOCKING, BRANCH. | BRANCH is a virtual step, representing an execution branch such as a lane. |
| updateTime | Long | Time of the last update to this step's information | |
| transitionMessage | String | Same as transitionDescription, or same as transitionName if no description was set (both are in the StepTransitionLog object). | |

**StepTransitionLog object:**

This object is one of the attributes of a step. It represents all information about the step's response and outgoing transition.

| Attribute | Type | Description | Comments |
|---|---|---|---|
| transitionName | String | Outgoing transition name. | |
| transitionDescription | String | Outgoing transition description (if one was set in Studio). | |
| responseName | String | Response name (given by the step's inner operation/flow). | |
| responseType | Predefined Value | Response type, can be one of: RESOLVED, ERROR, DIAGNOSED, NO_ACTION_ TAKEN, EXCEPTION. | EXCEPTION means the step's execution could not be completed. |
| transitionValue | Double | Outgoing transition's ROI value. | |

**Full step object:**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| stepInfo | Object | Represents a summary of the main step details. See StepInfo object. | |
| stepTransitionLog | Object | Represents all information about this step's response and outgoing transition. See StepTransitionLog object. | Only relevant to steps that have outgoing transitions. |
| description | String | Unused. | |
| stepPrimaryResult | String | Primary result of the step's inner operation/flow. | |
| operationGroup | String | Name of the worker group which the worker executing this step belongs to. | Only relevant to operation steps. |
| errorList | List of strings | If unexpected errors have occurred during the step's execution, they will be listed here. | |
| stepInputs | List of StepInput objects | The step's bound inputs. Each StepInputobject has three fields: name (String), value (String) and termName (currently unused). | |
| stepResult | Map (String to String) | A name-to-value map of the step results (the results defined in the step properties). | |
| rawResult | Map (String to String) | A name-to-value map of the step's raw results (provided by the inner operation/flow). | |
| extraData | Map (String to String) | A name-to-value map of extra data (for example, subflow steps have a FLOW_ UUID entry). | |
| executionId | String | ID of the execution this step is a part of. | |
| status | Predefined Value | Step execution status, can be one of: RUNNING, COMPLETED, ERROR, PAUSED, CANCELED | |
| workerId | String | UUID of the worker which executed this step. | Only relevant to operation steps. |
| user | String | User name of the OO user account which owned the execution at the time this step was executed. | |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful. If there are no results, the result will be empty, but this is still OK. |
| 400 | Bad Request | If any of the arguments are invalid |
| 404 | Not Found | The requested execution was not found or is not viewable. |

**Response entity body:**

- **on success**: Returns a JSON array of steps (will be empty if there are no results). Each step is an object with the following format:

```
{
"stepInfo":{
"stepId":"93969d82-e4e7-48de-8b6f-bd3c7c9dec02", "stepName":"Chop",
"path":"0.0", "responseType":"RESOLVED", "startTime":1396539495159,
"endTime":1396539495162, "paused":false, "orderNumber":"0000000000", "flowName":"My Flow",
"flowId":"7407b97b-fd42-47d2-a670-498278882b81", "type":"OPERATION",
"updatedTime":1396539495252,
"transitionMessage":"success"
}, "stepTransitionLog":{
"transitionName":"success", "transitionDescription":null, "responseName":"success",
"responseType":"RESOLVED",
"transitionValue":"100.0"
}, "description":null, "stepPrimaryResult":"bee", "operationGroup":"RAS_Operator_Path",
"errorList":[

], "stepInputs":[
{
"name":"foo", "termName":null, "value":"bar"
},

{
"name":"Field_1", "termName":null, "value":"beer"
}
],
"stepResult":{ "step_result1":"example", "step_result2":"another example"
}, "rawResult":{
"response":"success", "FailureMessage":null, "Result":"bee", "TimedOut":"false",
"returnResult":"bee",
"Field_1":"beer"
}, "extraData":{

}, "executionId":"100337054", "status":"COMPLETED",
"workerId":"f61428ff-e55a-4942-8351-f5b6b0b99b14", "user":"anonymousUser"
}
        },
        "executionId":"10033705
        4",
        "status":"COMPLETED",
        "workerId":"f61428ff-e55a-4942-8351-
        f5b6b0b99b14", "user":"anonymousUser"

}
```

## Get a Single Execution Step

**Request**: `GET /executions/{executionId}/steps/{stepPath}`

**Description**: Returns data about a single step of an execution. For more details, see Get Execution Steps (with Filtering and Pagination).

**Request path variables:**

| Attribute | Description |
|---|---|
| executionId | Execution ID whose step to retrieve. |
| stepPath | The step's path in the execution tree. Step paths work as follows:<br>The paths of the steps at the top level of the execution are 0.0 for the 1st step, 0.1 for the 2nd and so on.<br>If the 2nd step (0.1) is a subflow, then the paths of the steps in that subflow will be 0.1.0, 0.1.1 and so on.<br>**Note**: Virtual steps such as lanes also have a path. |

**Example:**

`GET /executions/100300001/steps/0.7.0.13`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | Successful. |
| 400 | Bad Request | Any of the arguments are invalid. |
| 404 | Not Found | The requested step wasn't found in the given execution, or the execution is not viewable |

## Get Execution Steps as a CSV File

**Request**: `GET /executions/{executionId}/steps?mediaType=csv`

**Description**: Returns all executed steps for the given execution, as a CSV file.

**Request path variables:**

| Attribute | Description |
|---|---|
| `executionId` | Execution ID whose steps to retrieve. |

**Request parameters:**

The `mediaType` parameter must be set to `csv`. Otherwise, the request will be treated as Get Execution Steps (with Filtering and Pagination). There are no other parameters.

**Example:**

`GET /executions/100300001/steps?mediaType=csv`

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | Successful. If there are no results, the result will be empty, but this is still OK. |
| 404 | Not Found | If the execution cannot be viewed or does not exist. |
| 400 | Bad Request | If any of the arguments are invalid. |

## Execution Pauses

**Request**: `GET /executions/{executionId}/pauses`

**Description**: Retrieves current pauses for the given execution id.

**Request path variables:**

| Attribute | Description |
|---|---|
| executionId | The ID of the execution which the client wishes to retrieve its pauses. |

**Example:**

`GET /executions/100001/pauses`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | All requested pauses were returned. |
| 404 | Not Found | The provided execution ID doesn't exist. |

**Response entity body:**

An array which contains all the current pauses of the requested execution. Each element in the array represents a pause of a lane in the execution tree.

The returned array is not ordered and is empty if no pauses exist for the given execution id. There are five possible reasons for pauses, which can be differentiated by inspecting the `pauseReason` attribute.

**Note:** There are different return attributes between the different types:

**Reason 1: Input Required**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| pauseReason | Predefined Value | The value `INPUT_REQUIRED` | |
| pauseId | Long | An ID for the returned pause. | |
| executionId | String | The execution ID | |
| branchId | String | The ID of the branch were the pause has occurred. | null ID represents the main branch. |
| stepId | String | The UUID of the step in which the pause has occurred. | |
| stepName | String | The name of the step in the flow. | |
| requiredInputs | FlowInput | See Get Flow Inputs | |

**Example:**

```
[
    {
        "pauseId":101100014,
        "executionId":"10010035
        5", "branchId":null,
        "stepId":"fa351d72-d381-4159-b81b-
        5fff493f7e41", "stepName":"Parallel Split",
        "pauseReason":"INPUT_REQUIRED",
        "requiredInputs":[
            {
                "uuid":null,
                "name":"parallelInput",
                "valueDelimiter":null,
                "description":"Just
                checking",
                "encrypted":false,
                "multiValue":false,
                "mandatory":false,
                "sources":null,
                "type":"String",
                "validationId":null,
                "defaultValue":null
            }
        ]
    }
]
```

**Reason 2: Display**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| pauseReason | Predefined Value | The value DISPLAY | |
| pauseId | Long | An ID for the returned pause. | |
| executionId | String | The execution ID. | |
| branchId | String | The ID of the branch were the pause has occurred. | A null ID represents the main branch. |
| stepId | String | The UUID of the step in which the pause has occurred. | |
| stepName | String | The name of the step in the flow. | |
| title | String | The author's specified localized title for the prompt message that should be presented to the user. | |
| text | String | The author's specified localized message that should be presented to the user | |
| height | String | For future use. | Ignore this field. |
| width | String | For future use. | Ignore this field. |

**Example:**

```
[
    {
        "pauseId":101100010,
        "executionId":"100100312",
        "branchId":null,
        "stepId":"c12fc96a-f938-4a8c-81a1-5dbcc3b81e6d",
        "stepName":"Resolved : success",
        "pauseReason":"DISPLAY",
        "title":"Return Step title with input: value1",
        "text":"Return Step text:\ninput1: value1",
        "height":"0.0",
        "width":"0.0"
    }
]
```

**Reason 3: Gated Transition**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| pauseReason | Predefined Value | The value GATED_TRANSITION | |
| pauseId | Long | An ID for the returned pause. | |
| executionId | String | The execution ID. | |
| branchId | String | The ID of the branch were the pause has occurred. | A null ID represents the main branch. |
| stepId | String | The UUID of the step in which the pause has occurred. | |
| stepName | String | The name of the step in the flow. | |
| roleName | String | The role which the user is missing in order to pass the gated transition. | |
| userName | String | The user which is trying to pass through the gated transition. | |

**Example:**

```
[
    {
        "pauseId":101100008,
        "executionId":"100100274",
        "branchId":null,
        "stepId":"2ba782f2-ea84-4f66-936d-4226d4bfa134",
        "stepName":"UUID Generator",
        "pauseReason":"GATED_TRANSITION",
        "roleName":"AUDITOR",
        "userName":"anonymousUser"
    }
]
```

**Reason 4: Hand Off**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| pauseReason | Predefined Value | The value HAND_OFF | |
| pauseId | Long | An ID for the returned pause. | |
| executionId | String | The execution ID. | |
| branchId | String | The ID of the branch were the pause has occurred. | A null ID represents the main branch. |
| stepId | String | The UUID of the step in which the pause has occurred. | |
| stepName | String | The name of the step in the flow. | |

**Example:**

```
[
    {
        "pauseId":101100009,
        "executionId":"100100293",
        "branchId":null,
        "stepId":"70eaf376-72ca-4440-9f60-a743fcfa56b2",
        "stepName":"UUID Generator",
        "pauseReason":"HAND_OFF"
    }
]
```

**Reason 5: No Workers In Group**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| pauseReason | Predefined Value | The value NO_WORKERS_IN_GROUP | |
| pauseId | Long | An ID for the returned pause. | |
| executionId | String | The execution ID | |
| branchId | String | The ID of the branch were the pause has occurred. | A null ID represents the main branch. |
| stepId | String | The UUID of the step in which the pause has occurred. | Would be null, please ignore |
| stepName | String | The name of the step in the flow. | Would be null, please ignore |
| groupName | String | The unavailable group. | |

**Example:**

```
[
    {
        "pauseId":101100012,
        "executionId":"100100341",
        "branchId":null,
        "stepId":null,
        "stepName":null,
        "pauseReason":"NO_WORKERS_IN_GROUP",
        "groupName":"RAS_Operator_Path"
    }
```

```
        ]
```

# Get Execution Log

**Request**: `GET /executions/{executionId}/execution-log`

**Description**: This API retrieves the extended summary of a specific execution. It is an extension of the Execution Summary API and holds additional information, such as the inputs and outputs of that execution.

**Request path variables:**

| Attribute | Description |
|---|---|
| executionId | The id of the execution |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | The requested execution log was. |
| 403 | Forbidden | |
| 404 | Not Found | The requested execution log was not found or not viewable. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
{
    "executionSummary":{
        "executionId":"348246628680024354",
        "branchId":null,
        "startTime":1371366300297, "endTime":null,
        "status":"PAUSED",
        "resultStatusType":"RESOLVED",
        "resultStatusName":"HAHA",
        "pauseReason":"USER_PAUSED",
        "owner":"anonymous",
        "ownerDomain:null",
        "triggeredBy":"anonymous",
        "flowUuid":"a8e8fc10-b584-4d39-921f-987b29c9dd19",
        "flowPath":null,
        "executionName":"mock flow",
        "triggeringSource:central"
        "roi":null
},
    "executionLogLevel": "STANDARD",
```

```
            "flowVars":[
                {
                    "name":"flowVar0",
                    "termName":"flowVar0TermName
                    ", "value":"flowVar0Value"
                },
                {
                    "name":"flowVar1",
                    "termName":"flowVar1TermName
                    ", "value":"flowVar1Value"
                },
                {
                    "name":"flowVar2",
                    "termName":"flowVar2TermName
                    ", "value":"flowVar2Value"
                },
                {
                    "name":"flowVar3",
                    "termName":"flowVar3TermName
                    ", "value":"flowVar3Value"
                },
                {
                    "name":"flowVar4",
                    "termName":"flowVar4TermName
                    ", "value":"flowVar4Value"
                }
            ],
            "flowOutput":{
                "flowOutput4":"flowOutput4Value
                ",
                "flowOutput3":"flowOutput3Value
                ",
                "flowOutput0":"flowOutput0Value
                ",
                "flowOutput2":"flowOutput2Value
                ",
                "flowOutput1":"flowOutput1Value
                "
            }
        }
    }
```

**Response Attributes:**

| Attribute | Type | Description |
|---|---|---|
| executionSummary | Predefined Value | An array which contains all of the executions summary which match the requested filters. |
| flowVars | List of FlowInput objects | The flow's bound inputs. Each FlowInput object has three fields: name (String), value (String) and termName (domain term that the input is recorded under). |
| flowOutput | Map (String to String) | A name-to-value map of the flows outputs. |

## Change the Status of an Execution

**Request**: `PUT /executions/{executionIds}/status`

**Description**: Update existing executions statuses. Each request has a single action and data but can be applied to multiple executions.

**Note**:

Changing the run status is allowed when both of these conditions are true:

- The user has the **Run** permission (entitlement) for the executed flow.
- The run is currently assigned to the user, or alternatively, the user has the **Manage Others' Runs** permission.

**Request path variables:**

| Attribute | Description | Type |
|---|---|---|
| executionIds | The IDs of the executions which the user wants to update the statuses. Separated by a comma. | String |

**Request entity body:**

The execution status can be changed to one of the following states: `CANCEL`, `PAUSE`, `REASSIGN`, or `RESUME`. The desired status should be set in the action attribute.

**Examples:**

**To cancel executions:**

| Attribute | Type | Description | Required | Default |
|---|---|---|---|---|
| action | Predefined Value | The value CANCEL | Yes | N/A |

```
{
"action":"CANCEL"
}
```

**To pause executions:**

| Attribute | Type | Description | Required | Default |
|---|---|---|---|---|
| action | Predefined Value | The value PAUSE | Yes | N/A |

```
{
"action":"PAUSE"
}
```

**For reassigning executions to another user**:

| Attribute | Type | Description | Required | Default |
|---|---|---|---|---|
| action | Predefined Value | The value REASSIGN | Yes | N/A |
| data | Key value | Contains the key userName and the reassigned user as value. | Yes | |

```
{
"action":"REASSIGN",
"data":{
"userName":"John"
      }
}
```

**For resuming executions:**

| Attribute | Type | Description | Required | Default |
|-----------|------|-------------|----------|---------|
| `action` | Predefined Value | The value `RESUME` | Yes | N/A |
| `data` | Key value | Should contain two key-value pairs:<br>• `branchId` key for the branch that should be resumed. Null should be provided for the branch root.<br>• `input_binding` key is used to provide the required inputs for an execution which is waiting for inputs.The input_binding value must be an object using the format: {input_name : input_value} | Yes – but it can be empty and use the default values. | • `brachId` default value is null<br>• `input_binding` only required when resuming an execution which has the pauseReason INPUT_REQUIRED. See Execution Pauses |

**Resume without inputs:**

```
{
        "action":"RESUME",
        "data":{
                "branchId":null
        }
}
```

**Resume with inputs:**

```
{
        "action":"RESUME",
        "data":{
                "branchId":"f47ac10b-58cc-4372-a567-0e02b2c3d479:1",
                "input_binding":{
                        "Input 1":"VALUE2",
                        "Input 2":[
                                "VALUE1",
                                "VALUE2",
                                "VALUE3"
                        ],
                        "Input 3":null,
                        "Input 4":"434"
                }
        }
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful. If there are failures it will be reflected in the response body. |
| 400 | Bad request | If the request was wrong. For example, if reassign without username data. |

**Response entity body:**

• **on success**: Returns a JSON object with the following format:

```
[

{
  "executionId":"33321909610082830376",
  "executionName":"mock flow1",
  "result":"SUCCESS"
 },
 {

"executionId":"3098745634785645678",
  "executionName":"mock flow2",
  "result":"FAILED_ALREADY_COMPLETED"
 }
 ]
```

The result is predefined value.

Values are: SUCCESS, FAILED_ALREADY_PAUSED, FAILED_ALREADY_COMPLETED, FAILED_ALREADY_CANCELED, FAILED_ALREADY_RUNNING, FAILED_NOT_FOUND, FAILED_PENDING_PAUSE, FAILED_SYSTEM_FAILURE, FAILED_FORBIDDEN, FAILED_BAD_REQUEST;

## Execute a Flow by UUID

**Request**: `POST /executions`

**Description**: Executes a flow specified by UUID.

Request entity body**:**

| Attribute | Type | Description | Required? | Default |
|-----------|------|-------------|-----------|---------|
| `flowUuid` | String | Flow ID | Yes | |
| `runName` | String | Execution run name | No | Flow name |
| `logLevel` | String | Execution log level. The logLevel attribute receives one of these values: `STANDARD`, EXTENDED | No | Flow log level |
| `inputs` | String | Execution bound inputs | No | |

The body of this request must include a JSON object with the following format:

```
{
        "flowUuid":"8d52dfc3-1de5-48d4-9c2a-887718de4696",
        "runName": "run1",
        "logLevel": "STANDARD",
        "inputs":
                {
                    "input1":"value for input1",
                    .
                    .
                    .
                    "inputn":"value for inputn"
                }
}
```

- There is an option to change the group alias mapping during a specific flow execution, by specifying an input called `OO_ALIAS_GROUP_MAPPING`.

For example to map the group alias `newAlias` to the worker group `group name` provide the following input:

```
"inputs":
{
"OO_ALIAS_GROUP_MAPPING":"{\"newAlias\":\"group name\"}"
.
.
.
}
```

`inputs` and `runName`, are optional and can be omitted.

**Note**: If the flows was defined with inputs which are either set as prompt user or/and are required, the flow will pause.

`LogLevel` is also optional – if not specified, the default level for the given flow will be used.

The `logLevel` attribute receives one of these values: `STANDARD`, `EXTENDED`.

42

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | A new flow execution was created. |
| 400 | Bad Request | One of the arguments are not correct. |

**Response entity body:**

• **on success**: Returns a string which represents execution id.

In the response, the Location header will contain the REST request relative path for retrieving information about the steps executed so far.

```
/executions/8d52dfc3-1de5-48d4-9c2a-887718de4696/steps
```

See the Get Execution API for more information.

## Ad-hoc Flow Execution

**Request**: `POST /executions`

**Description**: Ad-hoc Flow execution enables executing a flow without the need to first deploy the flow by providing the AFL flow xml.

**Note**: This feature will work only when authentication is not enabled in Central.

**Note**: You can view the `afl_xml_schema.xsd` file, which describes the HPE OO flow AFL (Automation Flow Language). You can view this file in the online help, or download it from <online- help folder>/ Content/Resources/OtherDocs/. You can also download this file from HPLN, located in the Operations Orchestration 10.x folder in the Resources Tab.

**Request entity body:**

The body of this request must include a JSON object with the following format:

```
{
    "aflContent":"AFL Flow",

    "runName": "run1",

    "logLevel": "STANDARD",

    "inputs":

        {
            "input1":"value for input1",

            .

            .

            "inputn":"value for inputn"

        }

}
```

- There is an option to change the group alias mapping during a specific flow execution, by specifying an input called `OO_ALIAS_GROUP_MAPPING`.

For example to map the group alias `newAlias` to the worker group `group name` provide the following input:

```
        "inputs":
            {
                "OO_ALIAS_GROUP_MAPPING":"{\"newAlias\":\"group name\"}"
                .
                .
                .
            }
```

The `aflContent` must include a JSON encoded AFL flow. `inputs` and `runName`, are optional and can be omitted.

`logLevel` is also optional – if not specified, the system default level will be used.

The `logLevel` attribute receives one of these values: `STANDARD`, `EXTENDED`.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | A new flow execution was created. |
| 400 | Bad Request | |

**Response entity body:**

- **on success**: Returns a string which represents execution id.

44

In the response, the `Location` header will contain the REST request relative path for retrieving information about the steps executed so far.

`/executions/8d52dfc3-1de5-48d4-9c2a-887718de4696/steps`

## Flow Execution Status

The following are the possible values of the status attribute, which appears in the APIs:

| Status | Description |
|---|---|
| RUNNING | The flow execution is in progress. |
| COMPLETED | The flow has finished. In order to understand the result (for example, success or failure) use the resultStatusType attribute. |
| SYSTEM_ FAILURE | The execution failed due to an unexpected error in the system. |
| PAUSED | The flow execution paused. For pause reasons, see the Execution Pauses API. |
| PENDING_ PAUSE | A flow execution pause request was submitted, and the system is waiting for an action to complete in order to enter the pause state. |
| CANCELED | The flow execution was canceled by the user. |
| PENDING_ CANCEL | A flow execution cancel request was submitted, and the system is waiting for an action to complete in order to cancel the execution. |

## Flow Input

Defines for the client how an input should be presented to the end user.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| uuid | String | The UUID of this input. | null is possible |
| name | String | A unique name of this input. | |
| valueDelimiter | String | The expected delimiter in the value, in case this is a multiple value input. | null is possible |
| description | String | A localized description of this input, this provides more information to the user. | |
| encrypted | Boolean | Indicates whether this is a classified input. It is advised to mask the user input in the presentation. | |
| multiValue | Boolean | Indicates whether multiple values are expected. They will be delimited by the valueDelimiter attribute. | |
| mandatory | Boolean | Set if the user must provide this input. In this case the attribute is true and the user does not provide the required input, the operation which requests this input will fail. | |
| sources | Array | Suggested input values for the user. | null is possible |
| type | Predefined Value | String: Indicates that a free text input is expected.<br>SelectionList: User should choose value/values from the supplied sources. | |
| validationId | String | For future use. | Would be null. Please ignore this attribute. |
| defaultValue | String | A default value for this input. This is a hint for the UI and could be used by the user. | null is possible |

**Example from Central UI:**

input2, list_bool, list_char, multi_list_char are flow inputs.

**Example:**

```
[
    {
        "uuid":"8e1b1288-3f1a-45ef-b23b-cbdf21bb607b",
        "name":"input2",
        "valueDelimiter":",",
        "description":"please enter your email",
        "encrypted":false,
        "multiValue":false,
        "mandatory":false,
        "sources":null,
        "type":"String",
        "validationId":null,
        "defaultValue":null
    },
    {
        "uuid":"fdd88ec2-b76f-4aec-a5af-509549bd41fb",
        "name":"list_bool",
        "valueDelimiter":",",
        "description":"choose yes or no",
        "encrypted":false,
        "multiValue":false,
        "mandatory":false,

        "sources":[ "True",

            "False"
        ],
        "type":"SelectionList",
        "validationId":null,
        "defaultValue":null
    },
    {
        "uuid":"3406b528-a856-49d1-82b3-516b7c8243c5",
        "name":"list_char",
        "valueDelimiter":",",
        "description":"this is encrypted input",
        "encrypted":false,
        "multiValue":false,
        "mandatory":true,
        "sources":[
            "Shift_JIS",
            "EUC-JP",
            "UTF-8",
            "UTF-32",
            "ISO-2022-JP",
            "UTF-16",
            "Windows-31J"
        ],
        "type":"SelectionList",
        "validationId":null,
        "defaultValue":null
    },
    {
        "uuid":"c4bcf870-a7f9-4160-8be0-eea2fc4978d4",
        "name":"multi_list_char",
        "valueDelimiter":",",
        "description":"this is multi-select input",
        "encrypted":false,
```

```
        "multiValue":true,
        "mandatory":false,
        "sources":[
            "Shift_JIS",
            "EUC-JP",
            "UTF-8",
            "UTF-32",
            "ISO-2022-JP",
            "UTF-16",
            "Windows-31J"
        ],
        "type":"SelectionList",
        "validationId":null,
        "defaultValue":null
    }
]
```

## Purge Debugger Execution Events

**Request**: `DELETE /debugger-events`

**Description**: Purges debugger events of remote executions.

This command is useful for keeping the database clean from debugger events, which become useless after executions have completed and also affect the size and memory the database occupies.

**Request parameters:**

| Attribute | Type | Description | Required | Default |
|-----------|------|-------------|----------|---------|
| endedBefore | Long | Time to use as upper limit for purging | Yes | |
| maxAmount | Long | Max number of executions to purge | No | 100 |

**Example:**

`DELETE /debugger-event?endedBefore=1411998175833`

`DELETE /debugger-event?endedBefore=1411998175833&maxAmount=1000`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful. If there are no results, the result will be empty (which is OK). |
| 400 | BadRequest | If any of the arguments are invalid. |
| 403 | Forbidden | The user attempting to execute this command does not have the Manage Data Cleanup permission. |

**Response entity body:**

- **on success**: returns the number of purged executions.

# Purge Execution Step Data, Inputs and Outputs

**Request**: `DELETE /executions`

**Description**: Purges execution data for runs matching the given criteria. It is possible to purge all execution data (including the execution summaries), or only specific items (flow inputs, flow outputs and/or steps).

**Request parameters:**

| Attribute | Type | Description | Required | Default value |
|---|---|---|---|---|
| `endedBefore` | Long | Time to use as upper limit for purging | Yes | |
| `maxAmount` | Integer | Max number of executions to purge | No | 100 |
| `purgeItems` | Predefined Value | List of items that should be purged.<br>Valid values: flowInputs, flowOutputs, steps.<br>If not specified, will purge all data for executions (including execution summaries). | No | |
| `statuses` | Predefined Value | List of execution statuses according to which the relevant execution will be purged.<br>Valid values:<br>`COMPLETED, COMPLETED_RESOLVED, COMPLETED_DIAGNOSED,COMPLETED_ERROR, COMPLETED_NO_ACTION_TAKEN, COMPLETED_CUSTOM SYSTEM_FAILURE, CANCELED.`<br>If not specified, will not filter according to status. | No | |
| `flowUuids` | String | Purge data for executions of the given flow UUIDs.<br>If not specified, will not filter according to flow UUID. | No | |

**Examples:**

- `DELETE /executions?endedBefore=1445421975041&flowUuids=2da44b9a-2161-47c5-94a9-864920bedef8`
  Purges all run data (execution summaries, steps, flow inputs and outputs) for runs of the flow with UUID 2da44b9a-2161-47c5-94a9-864920bedef8 which ended before 2015-10-21 10:06:15 UTC. Up to 100 runs will be purged.

- `DELETE /executions?endedBefore=1445421975041&statuses=SYSTEM_FAILURE,CANCELED &flowUuids=2da44b9a-2161-47c5-94a9-864920bedef8`
  Purges all run data (execution summaries, steps, flow inputs and outputs) for runs with status `SYSTEM_FAILURE` or `CANCELED` which ended before 2015-10-21 10:06:15 UTC. Up to 100 runs will be purged.

- `DELETE /executions?endedBefore=1412150176345&maxAmount=200&purgeItems=steps`
  Purges step data only, for runs which ended before 2014-10-01 07:56:16 UTC. Up to 200 runs will be purged.

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | Successful. The requested contents were deleted. |
| 400 | Bad Request | If any of the parameters are invalid. |
| 403 | Forbidden | The user attempting to execute this command does not have the ManageData Cleanup permission. |

**Response entity body:**

- **on success**: Returns the amount of executions whose summaries/steps/inputs/outputs were purged.

## Delete StepLog data

**Request**: `DELETE /steps-log`

**Description**: Purge step data according to time and amount of executions to purge.

**Example**: `DELETE /steps-log?endedBefore=1412150176345&maxAmount=200`

**Request parameters:**

| Attribute | Type | Description | Default value | Required? |
|-----------|------|-------------|---------------|-----------|
| `endedBefore` | Long | The time in milliseconds to start purging | 30 days back (in milliseconds) | No |
| `maxAmount` | int | The max amount of executions on which step's data is purged | 100 | No |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The step data was deleted successfully. |
| 400 | Bad Request | Arguments are invalid |
| 403 | Forbidden | The user does not have the permission Manage Cleanup Data |

**Response entity body:**

- **on success**: Returns a JSON string of the number of executions that their step data has been purged.

# Flow Library

APIs relating to the Flow Library

## Get Flows Library

**Request**: GET /flows/library

**Description**: Retrieves all flows deployed in the system. This returns an ordered list, and contains all the folders and flows in the flow library, ordered according to the tree structure.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested library was returned. |

**Response entity body:**

A List of elements which contains all of the flows that exist in the system.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| id | String | The id of node. | For a non-leaf (folder) node, the id is the same as the path. For a leaf node (deployed entity), the id is the entity's id. |
| name | String | The name of the node. | Folder name or flow name. |
| parentId | String | The id of the parent node. | Null if this is a root node. |
| leaf | Boolean | Whether the node is a leaf or not. | leaf is a flow, not leaf = folder |
| path | String | The path of the node. | |
| runnable | Boolean | Whether the node is a runnable or not. | |
| childrenIds | Array of Strings | List of the the node's children id | |

Remove this and replace

**Example:**

[{"id":"Library/cp-dotnet/InputsOutputsResponses/flows",
"name":"flows",
"parentId":"Library/cp-dotnet/InputsOutputsResponses",
"leaf":false,
"path":"Library/cp-dotnet/InputsOutputsResponses/flows",
"runnable":false,
"childrenIds":["Library/cp-dotnet/InputsOutputsResponses/flows/DOTNET_Action  inputs  and
outputs  2  op.xml","Library/cp-dotnet/InputsOutputsResponses/flows/DOTNET_Action  inputs
and outputs.xml"]},


{"id":"0e98f028-3a59-4cce-9196-4690bd48ea95",
"name":"DOTNET_Action inputs and outputs 2 op",
"parentId":"Library/cp-dotnet/InputsOutputsResponses/flows",
"leaf":true,
"path":"Library/cp-dotnet/InputsOutputsResponses/flows/DOTNET_Action inputs and
outputs 2 op.xml",
"runnable":true, "childrenIds":[]}]

**Example:**

## Read Next Level of Library Tree

**Request**: `GET /flows/tree/level`

**Description**: Returns a flat list of all tree Items under the path (lazy loading).

Request parameters:

| Attribute | Description | Required |
|-----------|-------------|----------|
| `path` | Return the tree items under this path. Default value is root. | No |

Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested items were found |

Response entity body:

- **on success:** Returns a JSON object with the following format:

```
[
    {
        "id":"library/Accelerator Packs",
        "name":"Accelerator Packs",
        "leaf":false,
        "path":"Library/Accelerator Packs",
        "runnable":false,
        "children":null
    },
    {
        "id":"library/How Do I flows",
        "name":"How Do I flows",
        "leaf":false,
        "path":"Library/How Do I flows",
        "runnable":false,
        "children":null
    }
]
```

## Get Flow Details

**Request**: `GET /flows/{uuid}`

**Description**: Returns flow properties by the uuid.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| uuid | The flow uuid |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested flow was found. |
| 404 | Not Found | The requested flow was not found or the user does not have permission to view it or the uuid was empty |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
{
    "id":"1fe1be31-2c78-40dd-8326-b8ca527e5587",
    "name":"Recently Run",
    "path":"Library/Utility Operations/Date and Time/Recently Run.xml",
    "description":"flow desciption",
    "cpName":"HPEOO-oo-base", "version":"version111"
    "logLevelInfo":{
      "logLevel":"EXTENDED",
      "logLevelSource":"SYSTEM"
     }
}
```

## Get Flow Inputs

**Request**: `GET /flows/{uuid}/inputs`

**Description**: Retrieves a list of flow's inputs by its UUID.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| uuid | The flow uuid |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested flow's inputs were found |
| 404 | Not Found | The requested flow was not found or the user does not have permission to view it or the uuid was empty |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
[
    {
        "uuid":"c4454566-6bb5-4be9-9824-2a08945f1574",
        "name":"message",
        "valueDelimiter":",",
        "description":"",
        "encrypted":false,
        "multiValue":false,
        "mandatory":true,
        "sources":null,
        "type":"String",
        "validationId":null,
        "defaultValue":null
    },
    {
        "uuid":"cdac00b3-f550-4cd5-a3eb-f15d2f80fd78",
        "name":"title",
        "valueDelimiter":",",
        "description":"",
        "encrypted":false,
        "multiValue":false,
        "mandatory":false,
        "sources":null,
        "type":"String",
        "validationId":null,
        "defaultValue":"Status message"
    }
]
```

## Get Flow Outputs

**Request**: `GET /flows/{uuid}/outputs`

**Description**: Retrieves a list of all the outputs defined for the flow with the requested uuid. The list of outputs retrieved will contain only outputs defined in the flow properties.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| uuid | The requested flow's uuid. |

**Example:**

`GET /flows/22de91c4-f651-42fd-a404-1bf0ca921f36/outputs`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested flow's inputs were found. |
| 404 | Not Found | The requested flow was not found or the user does not have permission to view it or the uuid was empty. |

**Response entity body:**

An array which contains all of the outputs that are defined for the flow. Each element in the array represents an output of the flow.

The returned array is not ordered.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| name | String | The name of the output. | |

**Example:**

```
[
{ "name":"Result"
},
{ "name":"output2"
},
{ "name":"output1"
}

]
```

## Get Flow Settings

**Request**: `GET /flows/{UUID}/settings`

**Description**: Retrieves execution settings for the specified flow.

**Request path variables:**

| Attribute | Description |
|---|---|
| UUID | The flow UUID |

Example:

`GET /flows/8aff6b34-15a2-11e5-b60b-1697f925ec7b/settings`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The flow settings were found |
| 404 | Not Found | The specified flow does not exist or the user is not entitled to view it |

**Response entity body:**

| Attribute | Type | Description |
|---|---|---|
| logLevelInfo | Object | The object contains two strings:<br><br>logLevel: The run log level (persistence level) of this flow.<br>Possible values: STANDARD, EXTENDED<br><br>logLevelSource: Indicates whether the log level is taken from the system default, or is explicitly set for this flow.<br>Possible values: SYSTEM, FLOW |
| flowTimeout | Integer or null | Possible values:<br><br>• null: This flow has no explicit timeout setting. The system default timeout will be used instead (see Get General Setting by Key).<br>• 0: This flow is explicitly set to have no timeout.<br>• Positive number: This flow is explicitly set to have the specified execution timeout, in minutes. |

**Example:**

```
{
      "logLevelInfo":{
            "logLevel": "STANDARD",
            "logLevelSource": "SYSTEM"
      },
      "flowTimeout": null
}
```

## Update Flow Settings

**Request**: `PUT /flows/{UUID}/settings`

**Description**: Sets execution settings for the specified flow.

Request path variables:

| Attribute | Description |
|-----------|-------------|
| UUID | The flow UUID |

Request entity body:

See response entity body if

| Attribute | Type | Description |
|-----------|------|-------------|
| logLevelInfo | Object | This object should contain a single property named `logLevel`.<br>Possible values for `logLevel`:<br>• `null`: Remove the explicit log level for this flow (use the system default log level).<br>• One of the predefined `logLevel` values (see Get Flow Settings). |
| flowTimeout | Integer or null | See possible values for `flowTimeout` in Get Flow Settings. |

**Example:**

```
PUT /flows/8aff6b34-15a2-11e5-b60b-1697f925ec7b/settings

{
        "logLevelInfo":{
                "logLevel": "STANDARD"
        },
        "flowTimeout": 60

}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 204 | Successful (No Content) | The flow settings were updated. |
| 400 | Bad Request | The request body is invalid. |
| 403 | Forbidden | The user does not have the Manage Content Packs permission. |
| 404 | Not Found | The specified flow does not exist or the user is not entitled to view it. |

# Scheduler

The scheduler API allows you to schedule flow executions. You can specify a schedule to run for a specific occasion. You can also setup recurring schedules for a flow for a repeated task. These APIs enable you to manage schedules, for example create new schedules.

**Note**: The scheduler supports only the quartz CRON syntax.

## Create New Flow-Schedule

**Request**: `POST /schedules`

**Description**: Add a new schedule for a flow execution.

**Request entity body:**

The body of this request must include a JSON object with the following format:

**JSON for a scheduled flow with a CRON triggering expression**:

```
{
        "flowScheduleName":"Scheduled Flow Created By REST",
        "flowUuid":"c34de7d6-14cc-4a1c-b25e-85afbb064359",
        "triggerExpression":"0 10 10 ? * 6",

        "startDate": 1314079869000,
        "endDate": 1491302669536,
        "runLogLevel": "STANDARD",
        "username": "DavisJ",
        "inputPromptUseBlank": true,
        "timeZone": "Asia/Amman",


        "inputs":
                {
                   "input1":"value for input1",
                    .
                    .
                    .
                   "inputn":"value for inputn"
                }
}
```

**JSON for a scheduled flow with a simple triggering expression:**

```
    {
        "flowScheduleName": "Scheduled Flow Created By REST",
        "flowUuid":"c34de7d6-14cc-4a1c-b25e-85afbb064359",
        "triggerExpression":"*/60000",
        "startDate": 1314079869000,
        "endDate": 1491302669536,
        "username": "DavisJ",
        "numOfOccurrences": 5,
        "runLogLevel": "STANDARD",
        "timeZone": "Asia/Amman",
        "inputPromptUseBlank": false,
        "inputs":{
            "input1": "value for input1",
            .
            .
            .
            "inputn": "value for inputn"
        }
    }
```

- If `endDate` is not set, by default, it receives a value of 0.

- If `username` is not set, by default, it receives a value of null.

- The trigger expression should be either a valid `cron` expression or a simple expression according the pattern below.

- If you use the `cron` expression, you can validate it using an <u>expression validity</u>.

- If you want to use a simple trigger expression (every x minutes) you should use the syntax according to the following example:

  `*/6000 = run every 60000 milliseconds (every minute)`

**Note**: If you use a `cron` expression you cannot add the `numOfOccurences` attribute as it may conflict with the `cron` expression. In addition, if you use simple triggers and add both end time and number of occurrences, the triggering ends according to the number of occurrences.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | A schedule was created successfully. |
| 400 | Bad Request | The user does not have Manage Schedules permission. |
| 403 | Forbidden | |

**Response entity body:**

- **on success**: Returns a JSON object of the created schedule with the following format:

```
{
"id": "1347298851037",

"flowScheduleName": "Scheduled Flow Created By REST",
"flowUuid": "c34de7d6-14cc-4a1c-b25e-85afbb064359",
"triggerExpression": "*/60000",
"startDate": 1314079869000,
"endDate": 1491302669536",

"username": "DavisJ",

"numOfOccurrences": 5,
"runLogLevel": "STANDARD",
"timeZone": "Asia/Amman",
"nextFireTime": -1,
"prevFireTime": -1,

"enabled": false,
"inputPromptUseBlank": false,

        "inputs":{
            "input1": "value for input1",
            .
            .
            .
            "inputn": "value for inputn"
        }
    }
```

In addition, a location header containing a URI to retrieve the created schedule for example:

```
/schedules/1347298851037
        {
            "flowScheduleName": "Scheduled Flow Created By REST",
            "flowUuid": "c34de7d6-14cc-4a1c-b25e-85afbb064359",
            "triggerExpression": "*/60000",
            "startDate": 1314079869000,
            "endDate": 1491302669536,
            "username": "DavisJ",
            "numOfOccurrences": 5,
            "runLogLevel": "STANDARD",
            "timeZone": "Asia/Amman",
            "inputPromptUseBlank": true,
            "inputs":{
                "input1": "value for input1",
                .
                .
                .
                "inputn": "value for inputn"
            }
        }
```

## Enable or Disable Flow-Schedule

**Request**: `PUT /schedules/{ids}/enabled`

**Description**: Enable or disable existing flow-schedules.

**Request path variables:**

| Attribute | Description |
|---|---|
| `ids` | The identifiers of the flow-schedules to enable or disable. |

**Request entity body:**

true or false

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | Successful (No Content) | The flow-schedules were updated successfully. |
| 400 | Bad Request | |
| 403 | Forbidden | The user does not have Manage Schedules permissions. |

**Request entity body**:

The body of this request needs to include a JSON value of either true to enable the schedules or false to disable them.

## Delete Flow-Schedule

**Request**: `DELETE /schedules/{ids}`

**Description**: Deletes flow-schedules according to the specified IDs.

**Request path variables:**

| Attribute | Description |
|---|---|
| ids | The identifiers of the flow-schedules to delete. |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | All specified flow-schedules were either deleted successfully, or did not exist to begin with. |
| 400 | Bad Request | |
| 403 | Forbidden | The user does not have Manage Schedules permissions. |

**Response entity body:**

- **on success:** Returns a JSON array of the deleted schedule IDs

## Get Flow-Schedules

**Request**: `GET /schedules`

**Description**: Returns all existing flow-schedules headers.

Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The requested flow-schedules were found. |
| 403 | Forbidden | The user does not have View Schedules or Manage Schedules permission. |

Request parameters:

| Attribute | Type | Description | Is required? | Default value |
|-----------|------|-------------|--------------|---------------|
| `start` | Integer | The page number to return (starting with 1 for the first page of results). | No | `1` |
| `pageSize` | Integer | The number of results in each page. | No | `50` |
| `direction` | Predefined Value | The direction in which to sort the results: `ASC` for ascending or `DESC` for descending. | No | `ASC` |
| `orderBy` | Predefined Value | The schedule property by which to sort the results: `scheduleName`, `flowName`, `nextFireTime`, `prevFireTime`, `scheduleState` or `username`. | No | `scheduleName` |
| `filter` | String | String to filter by. | No | `No filter` |

Example:

```
GET /schedules?start=3&pageSize=20&direction=DESC&orderBy=nextFireTime&filter=my-schedule
```

Response entity body:

- **on success:** Returns a JSON array, containing all existing flow-schedules headers, with the following format:

```
[
    {
            "id":"123",
            "enabled":true,
            "flowUuid":"78bec456-db6a-4c05-99ad-0675b230bfeb",
            "nextFireTime":-1,
            "prevFireTime":-1,
            "flowScheduleName":"schedule 1",
            "flowName":"flow1",
            "flowPath":"path0",
            "username":"admin"
            "triggerExpression":"0 10 10 ? * 6"
    },
    .
    .
    .
```

66

```
        .
        {
                "id":"567",
                "enabled":true,
                "flowUuid":"3d32e475g-ab54-fe21-df32-4743346ebebd",
                "nextFireTime":-1,
                "prevFireTime":-1,
                "flowScheduleName":"schedule n",
                "flowName":"flow3",
                "flowPath":"path2",
                "username":"admin"
                "triggerExpression":null
        }
]
```

## Get Flow-Schedule Details

**Request**: GET /schedules/{id}

**Description**: Returns details about a flow-schedule specified by ID.

**Request path variables:**

| Attribute | Description |
|---|---|
| id | The identifier of the flow-schedule to retrieve. |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The requested flow-schedule was found. |
| 403 | Forbidden | The user does not have View Schedules or Manage Schedules permission. |
| 404 | Not Found | The requested flow-schedule was not found. |

**Response entity body:**

- **on success:** Returns a JSON object of the flow-schedule details with the following format:

```
{
    "id": "1399455773960",
    "flowScheduleName": "Scheduled Flow Created By REST",
    "flowUuid": "c34de7d6-14cc-4a1c-b25e-85afbb064359",
    "triggerExpression": "0 10 10 ? * 6",
    "startDate": 1399455780000,
    "endDate": 1491302669536,
    "numOfOccurrences": null,
    "timeZone": "Asia/Amman",
    "username": "admin",
    "runLogLevel": "STANDARD",
    "nextFireTime": 1399619400000,
    "prevFireTime": -1,
    "enabled": true,
    "inputPromptUseBlank": false,
    "inputs":{
        "input1": "value for input1",
        "input2": "value for input2"
    }
}
```

68

## Update Flow-Schedule

**Request**: PUT /schedules/{id}

**Description**: Updates an existing flow-schedule. Includes a list of values that can be updated.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| id | The identifier of the flow-schedule to update. |

**Request entity body:**

The body of this request needs to include a JSON object with the following format:

```
{
    "flowScheduleName": "Scheduled Flow Created By REST",
    "flowUuid": "c34de7d6-14cc-4a1c-b25e-85afbb064359",
    "triggerExpression": "0 10 10 ? * 6",
    "startDate": 1376072040000,
    "endDate": 1377334800000,
    "runLogLevel": "STANDARD",
    "timeZone": "Asia/Amman",
    "inputPromptUseBlank": false,
    "inputs":{
        "input1": "value for input1",
        .
        .
        .
        "inputn": "value for inputn"
    }
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | Returned when update schedule successfully |
| 400 | Bad Request | |
| 403 | Forbidden | The user does not have Manage Schedules permission. |

**Response entity body:**

- **on success:** Returns a JSON value: true

# Dashboard

The Dashboard workspace reflects the system's ROI, and analyzed flow aggregation. It provides statistical information about the system (popular flows, result distribution, execution time, and so on) and financial information about the return on investment. This API allows you to get the statistic information in order to generate the reports for analyzing information.

## Get Statistics

**Request**: GET /executions/statistics

**Description**: Returns a flows statistic info (list of FlowStatisticsDataVO): ROI, number of executions, average execution time and result distribution.

### Request parameters:

| Attribute | Description | Type | Required |
|-----------|-------------|------|----------|
| top | Get top N results | Integer | No |
| measurements | Which statistics to display. If nothing is set then the four statistics are displayed. The following options are available: roi, numOfExecutions, avgExecutionTime, resultDistribution. | Predefined Value | No |
| sortBy | The following options are available: roi, numOfExecutions, avgExecutionTime<br>If nothing is set then:<br>• If the measurements list is empty, then the sort is set to numOfExecutions.<br>• If the Measurements are not empty, then nothing is sorted. If sortBy is set, then it should be contained in measurement (if supplied). | Predefined Value | No |
| sortDescending | Default is descending. | Boolean | No |
| endedBefore | Default is now. | Long | No |
| endedAfter | Default is one week ago. | Long | No |

### Example:

GET /executions/statistics?sortBy=avgExecutionTime&endedBefore=1415176455471&top=3

### Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | Operation was successful |
| 400 | Bad Request | • Wrong measurements value.<br>• Wrong sortByvalue. It must be included in the measurements, unless it's empty.<br>• Ended after > Ended Before |
| 403 | Forbidden | The user does not have dashboard read permission. |

**Response entity body:**

List<FlowStatisticsDataVO>:

```
[
  {
    "flowUuid" : 7f2f68-ef48-4a-4a91-bb11-5fadf44bebee",
    "flowPath" : "Library/Repo9x/Dev/ROI/ROI - Basic.xml",
    "flowRoi" : 30.0,
    "numberOfExecutions" : 3 "averageExecutionTime" : 11000,
    "resultsDistribution" : [
      {
        "type" : "RESOLVED",
        "amount" : 3
      }
    ]
  },
  {
    "flowUuid" : "0b6d9c00-2a34-4c63-8534-4a364d64272",
    "flowPath" : "Library/Repo9x/Dev/ROI/ROI - subflow.xml",
    "flowRoi" : 1830.0,
    "numberOfExecution" : 3,
    "averageExecutionTime" : 1000,
    "resultsDistribution" : [
      {
        "type" : "SYSTEM_FAILURE",
        "amount" : 2
      },
      {
        "type" : "RESOLVED",
        "amount" : 3
      }
    ]
  }
]
```

# Deployment

## Deploy Content Packs

**Request**: `PUT /content-packs/{cpFileName}`

**Description**: Deploys a content pack. The file extension should not be provided in the name.

### Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | Deployment succeed |
| 403 | Forbidden | The user does not have Manage Content Packs permission. |
| 417 | Expectation Failed | Deployment failed |

### Example:

`/content-packs/base-cp`

### Request path variables:

| Attribute | Description | Required |
|-----------|-------------|----------|
| name | The name of the content pack to be deployed. | Yes |

**Note**:

• The body of the request should contain the contents of the content pack file to be deployed.

### Response entity body:

• **on success:** Returns a JSON value: true

```
{
"aggregatedSeverity":"Info",
"contentPackResponses":{
"content pack file name.jar":{
"contentPackUUID":" a4618f99-309d-4537-b67c-e43dd9f73baa",
"contentPackName":"content pack filename.jar",
"message":"content pack file name.jar (author: date:)",
"responses":[
{
"contentPackName":"content pack file name.jar",
"responseCategory":"Success",
"level":"Info",
"message":"Successfully deployed content pack file name.jar"
}
]
}
}
}
```

The `aggregatedSeverity` and `level` attribute receives one of the following values: `Info`, `Warning`, and `Error`.

The `responseCategory` attribute receives one of the following values:

• `Success`: The content pack was deployed successfully.

• `ContentPackFile`: The content pack file was invalid.

• `FlowDependency`: Cannot deploy the content pack because of missing flow dependency.

• `OperationDependency`: Cannot deploy the content pack because of missing operation dependency.

• `Overwrite`: Cannot deploy the content pack because it can't overwrite the existed one because of flow/operation dependencies issues.

72

- `ScheduledFlow`: A list of scheduled flows that will be affected/deleted if the deployment will be carried out (since the deployment is trying to delete a flow that is scheduled to run).

- `Exception`: Cannot deploy the content pack because of an unexpected exception.

- **on error**: Returns a JSON value:

```
"aggregatedSeverity" : "Error",
"contentPackResponses" :{
        "cp.jar": {
         "contentPackUUID":"N/A",
                "contentPackName":"cp.jar",
                "message":"cp.jar (author: , date: )",
                "responses":[
                {
                        "contentPackName":"cp.jar",
                        "responseCategory":"FlowDependency",
                        "level":"Error",
                        "message":"Missing 'operation'/'flow' with UUID d1bbf441-824a-450e-afae-
                        2ddec0e0f35e, which is required by the 'flow':
                        'Library/tesdt/flowwww.xml'."
                }]
        }
    }
```

## How can I deploy content packs with progress?

1. Create Deployments – returns `deploymentProcessId`

2. Upload the Deployment Process File – to the given `deploymentProcessId`

3. Run a Specific Deployment Process – to the given `deploymentProcessId`

4. The next step is to request Get the Deployment Process Object. There are two kinds of responses:

      a.   `"status":"RUNNING"` – with data about the progress

      b.   `"status":"FINISHED"` – with data about the result

## Create Deployments

**Request**: `POST /deployments`

**Description**: Create a deployment process object and return the id to the client. You can also use this process for deleting content packs.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | Deployment progress object is created. |
| 403 | Forbidden | The user does not have Manage Content Packs permission. |

**JSON response example:**

```
{
    "deploymentProcessId":123
}
```

# Upload the Deployment Process File

**Request**: `POST /deployments/{deploymentProcessId}/files`

**Description**: Uploads one or more content pack files, and associates them with an existing deployment process (see Create Deployments). The request should be formatted according to the "multipart/form-data" standard for uploading files to a web server (RFC 2388).

**Note**: The request body should be the contents of the content pack files to be deployed.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `deploymentProcessId` | The deployment process ID to associate the files with. The ID is returned when creating a new deployment process. |

**Example:**

`POST /deployments/109600004/files`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The files were uploaded and associated with the specified deployment process. |
| 400 | Bad Request | Some or all of the uploaded files have failed basic validation. |
| 403 | Forbidden | The user does not have Manage Content Packs permission. |
| 404 | Not Found | The deployment process object with the specific id does not exist. |
| 409 | Conflict | Some or all of the uploaded content packs have already been added to this deployment process. |
| 500 | Internal Server Error | The HTTP request was not a valid "multipart/form-data" request (RFC 2388). |

**JSON response example:**

The response body includes details for each of the uploaded files. The `signDetails` property contains the content pack's digital signature details. The `deleteUrl` and deleteType properties represent a REST request, which you can send to remove the particular file from this deployment process.

```
{
  "files":
  [
    {
      "name":"cp-signed.jar",
      "size":10095,
      "fileUploadId":110800009,
      "deleteUrl":"/deployments/109600005/files/110800009",
      "deleteType":"DELETE",
      "signDetails":
        {"signStatus":"signed",
         "signedBy":"CN=qa-MAINDC-CA, DC=qa, DC=ad, DC=com",
         "warnings":[],
         "certs":
         [
           {
             "certType":"X.509",
             "certDn":"CN=Administrator, CN=Users, DC=qa, DC=ad, DC=com",
             "certKeystoreEntryAlias":null,
             "validityTimeFrom":1417599979000,
             "validityTimeNotAfter":1480758979000,
             "validityTimeNotBefore":0,
             "supportCodeSign":false
           },
           {
```

```
                    "certType":"X.509",
                    "certDn":"CN=qa-MAINDC-CA, DC=qa, DC=ad, DC=com",
                    "certKeystoreEntryAlias":null,
                    "validityTimeFrom":1390826569000,
                    "validityTimeNotAfter":1706359968000,
                    "validityTimeNotBefore":0,
                    "supportCodeSign":false
                }
            ],
          "trusted":false}
        }
      ]
    }
```

# Run a Specific Deployment Process / Delete Process

**Request**: PUT /deployments/{deploymentProcessId}

**Description**: Run the specific deployment process. Make sure that no other process is running.

**Request path variables:**

| Attribute | Description |
|---|---|
| deploymentProcessId | The requested deployment process id. The id is returned when creating a new deployment process. |

**Request parameters:**

| Attribute | Type | Description | Is required? | Default value |
|---|---|---|---|---|
| force | Boolean | Deploy/delete even when there are warnings (for example, scheduled runs for a flow being deleted). | No | false |

**Examples:**

PUT /deployments/12345?force=true

PUT /deployments/12345

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | No Content | The deployment process has started successfully. |
| 403 | Forbidden | The user who executed this command does not have permission to manage content packs. |
| 404 | Not Found | The requested deployment process is not found. |
| 409 | Conflict | Another deployment is in process. |
| 500 | Bad Request | The deployment process status is not PENDING or there are no content packs for deployment and no content packs for deleting, or there are both content packs for deployment and content packs for delete. <br> In addition: <br> • When this process is already running or finished. <br> • There are no content pack files for deleting. <br> • Both content pack files for deployment and content files for deleting exist together. |

## Get the Deployment Process Object

**Request**: `GET /deployments/{deploymentProcessId}`

**Description**: Gets the status and details of a previously-created deployment process. A deployment process is usually used when <u>deploying files with progress</u>, but can also be used for <u>deleting a content pack</u>.

**Example:**

`/deployments/3332190961082830376`

**Request path variables:**

| Attribute | Description |
|---|---|
| deploymentProcessId | The deployment process ID. The ID is returned when creating a new deployment process. |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The specific deployment process was found. |
| 403 | Forbidden | The user does not have Read Content Packs or Manage Content Packs permission. |
| 404 | Not Found | The deployment process object with the specific id does not exist. |

These are the responses:

- `status:RUNNING` – with data about the progress.

- `status:FINISHED` – with data about the result.

- `status:PENDING` – when the deployment was created and files are being uploaded, and before it has started running.

```
{
        "deploymentProcessId": 137100008,
        "status": "PENDING",
        "currentStep": "Downloading files ",
        "currentStepIndex": 0,
        "numOfSteps": 11,
        "numOfSubSteps": 0,
        "currentSubStep": 0,
        "deploymentResultVO": null
}
```

If your deployment process is still running, the JSON response appears as follows:

```
{
        "deploymentProcessId": 137100005,
        "status": "RUNNING",
        "currentStep": "Deploying artifacts",
        "currentStepIndex": 12,
        "numOfSteps": 16,
        "numOfSubSteps": 470,
        "currentSubStep": 409,
        "deploymentResultVO": null
}
```

If your deployment process has finished or failed, the JSON appears as follows:

```
{
"deploymentProcessId": 137100005,
"status": "FINISHED",
"currentStep": "Committing data",
"currentStepIndex": 16,
"numOfSteps": 16,
"numOfSubSteps": 1,
"currentSubStep": 0,
"deploymentResultVO": {
        "aggregatedSeverity": "Info",
        "contentPackResponses": {
                "fv11n-test-content-pack-10.10.182.jar": {
                 "contentPackUUID":" a4618f99-309d-4537-b67c-e43dd9f73baa",
                        "contentPackName": "fv11n-test-content-pack-10.10.182.jar",
                        "message": "fv11n-test-content-pack-10.10.182.jar (author: , date: )",
                        "responses": [{
                                "contentPackName": "fv11n-test-content-pack-10.10.182.jar",
                                "responseCategory": "Success",
                                "level": "Info",
                                "message": "Successfully deployed fv11n-test-content-pack-
                        10.10.182.jar"}]
                        }
                }
        }
}
```

## Delete Content Pack from Deployment Process

**Request**: `DELETE /deployments/{deploymentProcessId}/files/{fileId}`

**Description**: Delete a specific content pack from a deployment process.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 204 | No Content | The specific file was deleted from deployment process. |
| 403 | Forbidden | The user does not have Manage Content Packs permission. |
| 404 | Not Found | The deployment process object with the specific id does not exist. |

## Deleting Content Packs

To delete a content pack(s), you need to perform the following:

1. Create Deployments, returns the `deploymentProcessId`

   **Example**: `POST /deployments`

   **JSON response example**:

   ```
   {
     "deploymentProcessId":12345
   }
   ```

2. Enter the id of the content pack file that you want to delete, to the given `deploymentProcessId`. See Adding Content Pack Files for Deleting.

   **Example**: `POST /deployments/12345/cpsForDelete`

3. Run a Specific Deployment Process to the given `deploymentProcessId`.

   **Example**: `PUT /deployments/12345`

4. The next step is to request Get the Deployment Process Object.

## Adding Content Pack Files for Deleting

**Request**: `POST /deployments/{deploymentProcessId}/cpsForDelete`

**Description**: Add content pack files to the deployment process that you want deleted during the process.

**Request path variables:**

| Attribute | Description |
|---|---|
| `deploymentProcessId` | The requested deployment process id. The id is returned when creating a new deployment process. |

**Example:**

`POST /deployments/12345/cpsForDelete`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | No Content | The requested content packs were added to the deployment process. |
| 403 | Forbidden | The user who executed this command does not have permission to manage content packs. |
| 404 | Not Found | The requested deployment process is not found. |
| 500 | Bad Request | The requested content pack is not deployed or the deployment process status is not PENDING. |

**Request entity body:**

A string array which contains all of the ids of content packs that should be deleted. To acquire the content pack ID see Get Content Packs.

**Example:**

`["39d15573-aad6-44b3-a571-39c98c9bd508","a2c87d2a-2192-4087-8387-763d38246d26"]`

## Roll Back Last Deployment

**Request**: `DELETE /content-packs/last`

**Description**: Rolls back the last content pack deployment or deletion. Note that only the last action can be rolled back; you cannot roll back twice in a row.

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | Successful (no-content) | The last deployed content pack had been removed |
| 400 | Bad Request | There was already rollback on the last revision. |
| 403 | Forbidden | The user does not have Manage Content Packs permission. |
| 404 | Not Found | There is no deployment to rollback. |

# Content Packs

A content pack is a collection of flows, operations, configuration items (selection lists, system accounts, group aliases, and so on), as well as the binaries required to run actions. A content pack can be created in Studio by an author, or it can be provided by HPE or a third party.

The following table lists the metadata for a content pack:

| Attribute | Type | Description | Comments |
|---|---|---|---|
| id | String | The id of the content pack. | |
| name | String | The name of the content pack. | |
| version | String | The version of the content pack. | Can be null. |
| publisher | String | The name of the publisher that created the content pack. | Can be null. |
| description | String | The description of the content pack. | Can be null. |
| deploymentDate | Long | Time-stamp in milliseconds when the deployment of the content pack started. | Can be null. |
| signDetails | SignDetailsVO | The signature details of the content pack | |

**SignDetailsVO:**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| signStatus | String | The signature status, can be: signed, notSigned, broken, expired, na | |
| signedBy | String | The issuer distinguished name of the signature. | |
| warnings | List<String> | List of signing warnings | Can be empty. |
| certs | List< CertSignVO> | List of all signature's CAs. | Can be empty. |
| trusted | boolean | Whether the CA is trusted by OO or not (The CA should be imported into the client.truststore) | |

**CertSignVO**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| certType | String | The type of the certificate | |
| certDn | String | The subject of the certificate | |
| certKeystoreEntryAlias | String | The CA entry alias | |
| validityTimeFrom | long | The from date of the certificate validity | milliseconds |
| validityTimeNotAfter | long | The not after (end) date of the certificate validity | milliseconds |
| validityTimeNotBefore | long | The not before date of the certificate validity | milliseconds |
| supportCodeSign | boolean | True if the certificate is designed to be a code signer | |

## Get Content Packs

**Request**: `GET /content-packs`

**Description**: Retrieves a list of all the deployed content packs and with the related details.

**Request parameters:**

| Attribute | Type | Description | Is required? | Default value |
|---|---|---|---|---|
| `active` | Boolean | Allows you to retrieve either a list of all content packs including content (flows, operations and configuration items) and binaries or content packs including just content. | No | False |

**Example:**

```
GET /content-packs
```

```
GET /content-packs?active=false
```

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The requested content pack was returned. |
| 403 | Forbidden | The user who executed this command does not have permission to view content packs. |

**Response entity body:**

An array which contains all of the content packs which are deployed in the central.

Each element in the array represents a content pack. See the content pack meta-data table.

The returned array is ordered by descending deployment times, in other words, the content pack that was deployed last will be first.

**Example:**

```
[
{
  "name":"cp-with-only-success-flow",
  "version":"10.30.2",
  "id":"a4deec31-f10e-4249-9a71-7309fe0dbc5a",
  "publisher":"OO Tests",
  "description":"HPE OO cp-with-only-success-flow Test Content Pack",
  "deploymentDate":1418208680626,
  "deployedBy":"anonymousUser",
  "signDetails":
  {
  "signStatus":"signed",
  "signedBy":"CN=qa-MAINDC-CA, DC=qa, DC=ad, DC=com",
  "warnings":[],
  "certs":
  [
    {
      "certType":"X.509",
      "certDn":"CN=Administrator, CN=Users, DC=qa, DC=ad, DC=com",
      "certKeystoreEntryAlias":null,
      "validityTimeFrom":1417599979000,
      "validityTimeNotAfter":1480758979000,
      "validityTimeNotBefore":0,
      "supportCodeSign":false
    },
    {
      "certType":"X.509",
      "certDn":"CN=qa-MAINDC-CA, DC=qa, DC=ad, DC=com",
      "certKeystoreEntryAlias":null,
      "validityTimeFrom":1390826569000,
      "validityTimeNotAfter":1706359968000,
```

```
        "validityTimeNotBefore":0,
        "supportCodeSign":false
      }
    ],
    "trusted":false
    }
},
    Another one……
]
```

# Get Single Content Pack Details

**Request**: `GET /content-packs/{id}`

**Description**: Retrieves details of the specified deployed content pack.

**Request path variables:**

| Attribute | Description | Type |
|---|---|---|
| `id` | The requested content pack's id. The id may be acquired by retrieving the list of all content packs and extracting the id of element of the required content pack by its name. | int |

**Example:**

`GET /content-packs/b137e165-f4f7-4201-b262-2265c8085d27`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The requested content pack was returned. |
| 401 | Unauthenticated | The user did not authenticate and anonymous authentication is not allowed. |
| 403 | Forbidden | The user who executed this command does not have permission to view content packs. |
| 404 | Not Found | The requested content pack was not found. |

**Response entity body:**

A content pack element which represents the details of requested Content Pack. See the content pack meta-data table.

```
{
  "name":"cp-with-only-success-flow",
  "version":"10.30.2",
  "id":"a4deec31-f10e-4249-9a71-7309fe0dbc5a",
  "publisher":"OO Tests",
  "description":"HPE OO cp-with-only-success-flow Test Content Pack",
  "deploymentDate":1418208680626,
  "deployedBy":"anonymousUser",
  "signDetails":
  {
  "signStatus":"signed",
  "signedBy":"CN=qa-MAINDC-CA, DC=qa, DC=ad, DC=com",
  "warnings":[],
  "certs":
  [
    {
      "certType":"X.509",
      "certDn":"CN=Administrator, CN=Users, DC=qa, DC=ad, DC=com",
      "certKeystoreEntryAlias":null,
      "validityTimeFrom":1417599979000,
      "validityTimeNotAfter":1480758979000,
      "validityTimeNotBefore":0,
      "supportCodeSign":false
    },
    {
      "certType":"X.509",
      "certDn":"CN=qa-MAINDC-CA, DC=qa, DC=ad, DC=com",
      "certKeystoreEntryAlias":null,
      "validityTimeFrom":1390826569000,
      "validityTimeNotAfter":1706359968000,
      "validityTimeNotBefore":0,
      "supportCodeSign":false
    }
  ],
  "trusted":false
  }
}
```

## Get Content Pack Contents Tree

**Request**: `GET /content-packs/{id}/content-tree`

**Description**: Returns a flat list that of the contents of the content pack in tree-nodes format.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| id | The requested content pack's id. The id may be acquired by retrieving the list of all content packs and extracting the id of element of the required content pack by its name. |

**Example:**

`GET /content-packs/b137e165-f4f7-4201-b262-2265c8085d27/content-tree`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The requested content pack's contents were returned. |
| 403 | Forbidden | The user who executed this command does not have permission to view content packs. |
| 404 | Not Found | The requested content pack was not found. |

**Response entity body:**

An array of elements which represents one or two trees that contain all of the entities contained in the content pack.

Each element in the array represents a tree node which is either a folder (which isn't a leaf and has children) or a content element (which is a leaf and doesn't not have children).

The returned array is ordered in the following way: first the tree whose root is the "Library" folder, then the tree whose root is the "Configuration" folder. Each tree is ordered alphabetically by the path attribute.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| id | String | The id of node. | For a non-leaf (folder) node, the id is the same as the path. For a leaf node (deployed entity), the id is the entity's id. |
| name | String | The name of the node. | Folder name or deployed entity name. |
| parentId | String | The id of the parent node. | Null if this is a root node. |
| leaf | Boolean | Whether the node is a leaf or not. | |
| path | String | The path of the node. | |
| type | Predefined Value | The type of node. Possible values: `FOLDER, FLOW, OPERATION, CATEGORY, DOMAIN_ TERM, GROUP_ALIAS, ROLE_ALIAS, SCRIPTLET, SELECTION_LIST, SYSTEM_ACCOUNT, SYSTEM_ EVALUATOR, SYSTEM_FILTER, SYSTEM_PROPERTY` | |

**Example:**

```
[
{ "id":"Library",
"name":"Library",
"parentId":null,
"leaf":false,
"path":
"Library",
"type":"FOLDER"
},
{ "id":"Library/MyFlows",
"name":"MyFlows",
"parentId":"Library",
"leaf":false,
"path":"Library/MyFlows",
"type":"FOLDER"
},
{

"id":"a3e6572a-8268-4124-9ccb-b3d7e46e5051",
    "name":"my-flow1",
    "parentId":"Library/MyFlows",
    "leaf":true,
    "path":"Library/MyFlows/my-flow1.xml",
    "type":"FLOW"
    },
    {
    "id":"ddee425f-f259-4012-b799-51e0fa40151a",
    "name":"my-flow2",
    "parentId":"Library/MyFlows",
    "leaf":true,
    "path":"Library/MyFlows/my-flow2.xml",
    "type":"FLOW"
    },
    { "id":"Library/MyOperations",
    "name":"MyOperations",
    "parentId":"Library",
    "leaf":false,
    "path":"Library/MyOperations",
    "type":"FOLDER"
    },
    {
    "id":"17c8bb11-bbd8-4b1b-bfaa-5c3f1ef90051",
    "name":"operation1",
    "parentId":"Library/MyOperations", "leaf":true,
    "path":"Library/MyOperations/my-operation.xml",
    "type":"OPERATION"
    },
    { "id":"Configuration",
    "name":"Configuration",
    "parentId":null,
    "leaf":false,
    "path":"Configuration",
    "type":"FOLDER"
    },
    { "id":"Configuration/Categories",
    "name":"Categories",
    "parentId":"Configuration",
```

```
        "leaf":false,
        "path":"Configuration/Categories",
        "type":"FOLDER"
        },
        {
"id":"6a3a17c4-2c84-46c9-bef6-e043433878d6", "name":"my-category",

        "parentId":"Configuration/Categories",
        "leaf":true,
        "path":"Configuration/Categories/my-category.xml",
        "type":"CATEGORY"
        },
        {
        "id":"Configuration/Domain Terms",
        "name":"Domain Terms",
        "parentId":"Configuration",
        "leaf":false,
        "path":"Configuration/Domain Terms",
        "type":"FOLDER"
        },
        {
        "id":"2bc8c8e3-b742-43e5-8038-7c060dc8df77",
        "name":"my-domain-term",
        "parentId":"Configuration/Domain Terms",
        "leaf":true,
        "path":"Configuration/Domain Terms/my-domain-term.xml",
        "type":"DOMAIN_TERM"
        }
    ]
```

## Get Content Pack Statistics

**Request**: GET /content-packs/statistics

**Description**: Retrieves the content pack usage statistics according to the given request parameters.

**Notes**:

A content pack will be counted if it is actually used by the executed flow (as an operation or subflow). Each content pack will be counted at most once for every execution.

Once an execution ends, it will take 10 to 20 minutes to show up in the statistics.

### Request parameters:

| Attribute | Type | Description | Required |
|-----------|------|-------------|----------|
| cpNames | List of Strings | The content pack names to retrieve the statistics for. An empty list means all the content packs. Note that the content pack does not have to be currently deployed. | No |
| months | List of numbers | The months to retrieve the statistics for. An empty list means all months. | No |
| years | List of numbers | The years to retrieve the statistics for. An empty list means all years. | No |

### Example:

GET/content-packs/statistics?cpNames=Base,Solutions&months=1,2,3&years=2015

### Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The requested content pack statistics was returned. |
| 403 | Forbidden | The user who executed this command does not have permission to view or manage content packs. |

### Response entity body:

A map containing all of the content pack statistics that match the request.

The first level of the map is the statistics years. Each year is mapped to the different months, and each month contains the statistic element. This statistics element contains the total runs for that year and month and a map of content packs with their total runs.

### Example:

```
{
  "2014":{
    "12":{
      "total": 6500,
      "cps":{
        "Base": 6200,
        "Solutions": 1600,
        "Database": 300
      }
    }
  },
  "2015":{
    "1":{
      "total": 7000,
      "cps":{
        "Base": 7000,
        "Solutions": 2000,
        "Database": 1300
      }
    },
```

```
        "2":{
          "total": 1000,
          "cps":{
            "Base": 950,
            "Solutions": 400
          }
        }
      }
    }
```

# Get Content Pack Changes

**Request**: `GET /content-packs/{id}/changes`

**Description**: Retrieves all deployed entities changes from the last deployment of this content pack.

**Request path variables:**

| Attribute | Description |
|---|---|
| `id` | The requested content pack's id. The id may be acquired by retrieving the list of all content packs and extracting the `id` of element of the required content pack by its name. |

**Example:**

`GET /content-packs/b32b3a3d-0d7a-4780-85a1-5438987803ef/changes`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The requested content pack's changes were returned. |
| 403 | Forbidden | The user who executed this command does not have permission to view content packs. |
| 404 | Not Found | The requested content pack was not found. |

**Response entity body:**

An array which contains all of the deployed entities changes from the last deployment of this content pack.

Each element in the array represents a change to a deployed entity. The returned array is not ordered.

| Attribute | Type | Description | Comments |
|---|---|---|---|
| `id` | String | The id of the deployed entity. | |
| `changeType` | Predefined Value | The type of change that occurred. Possible values:<br>`ADD`: The deployed entity was added.<br>`DEL`: The deployed entity was deleted.<br>`MOD`: The deployed entity was modified (name changed, moved, xml changed, value changed) | |
| `oldCpName` | String | The name of the content pack that the entity belonged to before the last deployment. | Null if the content pack name hasn't changed. |
| `oldPath` | String | The path of the entity in the old content pack. | Null if the path has not changed. |
| `currentPath` | String | The current path of the entity. | |
| `entityType` | Predefined Value | The type of entity that changed. Possible values:<br>`FLOW, OPERATION, CATEGORY, DOMAIN_TERM,`<br>`GROUP_ALIAS, ROLE_ALIAS, SCRIPTLET, SELECTION_LIST, SYSTEM_ACCOUNT, SYSTEM_ EVALUATOR, SYSTEM_FILTER, SYSTEM_ PROPERTY` | |
| `xmlChanged` | Boolean | Whether the entity's xml was changed or not. | |

**Example**

```
[
{
"id":"c8100970-8044-4d70-abbe-72f1a4ace903",
"changeType":"ADD",
"oldCpName":null,
"oldPath":null,
"currentPath":"Library/cp-filters/Set of filters/ops/op - set of filters TC3
validation.xml",
"entityType":"OPERATION",
"xmlChanged":false},
{
"id":"e3aac28a-b430-47e9-b285-342fa7e2310f",
"changeType":"DEL",
"oldCpName":null,
"oldPath":"Library/my-flows/delete-this-flow.xml",
"currentPath":null,
"entityType":"FLOW",
"xmlChanged":false
},
{
"id":"1465b8aa-1ce6-4b2a-a3d3-d6214bd7b025",
"changeType":"MOD",
"oldCpName":"old-cp",
"oldPath":"Library/my-flows/subflows.xml",
"currentPath":"Library/events/Flow with subflows.xml",
"entityType":"FLOW",
"xmlChanged":false
}

]
```

# Configuration Items

## Get a Configuration Item (content)

**Request**: `GET /config-items/{type}/{path}`

**Description**: Returns a Configuration Item, such as a System Account.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| type | The type of the Configuration Item to return. Supported types: `domain-terms, group-aliases, selection-lists, system-accounts, system-properties` |
| path | The relative path of the Configuration Item. For example, if the item is stored at `Configuration/System Accounts/folder/sa.xml`, the relative path is `folder/sa`. **Note**: Paths are case-sensitive in Central. |

**Example:**

`GET /config-items/system-accounts/folder1/folder2/sa1`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful |
| 403 | Forbidden | The user is not allowed to view Configuration Items |
| 404 | Not Found | The Configuration Item does not exist, or the given type is not supported. For System Accounts, this can also mean the user does not have privileges to access this particular item. |

**Response body:**

• **on success**, returns a JSON object with the following properties:

| Attribute | Description | Type |
|-----------|-------------|------|
| type | The Configuration Item type, in the same format as given in the request URI (see above). | Predefined Value |
| path | The item's relative path, as given in the request URI (see above) | String |
| name | The item's name | String |
| value | The item's actual value, as will be used in flow execution. This will be the same as `customValue` if set, otherwise it will be the same as `defaultValue`.<br>• For Domain Terms and Selection Lists, the value is a pipe- delimited collection of values. For example: "`First Value\|Second Value\|Third Value`".<br>• For System Accounts, the value is formatted as a JSON object with properties username and password (see example below). Note that the server never sends passwords; they are masked as asterisks. | String |
| defaultValue | The item's default (deployed) value | String |
| customValue | The item's custom (override) value, which can be set in Central or through REST | String |
| fullPath | The item's full path in the Content Pack, including the .xml suffix | String |

| uuid | The item's UUID – will be null if this item is not deployed (or if it is a duplicate) | String |
|------|----------------------------------------------------------------------------------------|--------|

**Example:**

```
{
  "type":"system-accounts",
  "path":"folder1/folder2/sa1",
  "name":"sa1",
  "value":"{\"username\":\"admin\",\"password\":\"***********\"}",
  "defaultValue":"{\"username\":\"admin\",\"password\":\"***********\"}",
  "customValue":null,
  "fullPath":"Configuration/System Accounts/sa1.xml",
  "uuid":"4f32bcb8-969c-470f-9803-f823b72a9436"

}
```

```
{
  "type":"system-accounts",
```

## Get Configuration Items by type (content)

**Request**: `GET /config-items/{type}`

**Description**: Returns all Configuration Items of the specified type.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| type | The type of the Configuration Item to return. <br> For details, see Get a Configuration Item (content). |

**Example:**

`GET /config-items/system-accounts`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful |
| 403 | Forbidden | The user is not allowed to view Configuration Items |
| 404 | Not Found | The given type is not supported |

**Response body:**

On success, returns a JSON array of Configuration Items. The array will be empty if there are no items of the requested type.

For the format of each item, see Get a Configuration Item (content).

## Get All Configuration Items (content)

**Request**: `GET /config-items`

**Description**: Returns all Configuration Items of the supported types (see Get a Configuration Item (content) for a list of these types).

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful |
| 403 | Forbidden | The user is not allowed to view Configuration Items |

**Response body:**

Returns a JSON array of Configuration Items (will be empty if there are none). For the format of each item, see Get a Configuration Item (content).

## Set a Configuration Item's value (content)

**Request**: PUT /config-items/{type}/{path}

**Description**: Sets (or clears) the custom value of an existing Configuration Item, such as a System Account. If the item has a default (deployed) value, the custom value will override it.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| type | The type of the Configuration Item to return. <br> For details, see Get a Configuration Item (content). |
| path | The relative path of the Configuration Item. <br> For details, see Get a Configuration Item (content). |

**Example:**

PUT /config-items/system-accounts/folder1/folder2/sa1

**Request body:**

The body must include the new value as a JSON string. To clear the value, put null in the body.

**Examples:**

- "some value"

- "some value with \"quotes\" in it"

- null

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful |
| 400 | Bad Request | The given value is not valid JSON (must be either a string or null), or it is not formatted correctly for this Configuration Item's type |
| 403 | Forbidden | The user is not allowed to manage Configuration Items |
| 404 | Not Found | The Configuration Item does not exist, or the given type is not supported <br> For System Accounts, this can also mean the user does not have privileges to access this particular item. |

**Note**: Some Configuration Item types expect a particular value format. For details, see the response body section in Get a Configuration Item (content).

**Response body:**

- **on success**: returns the updated Configuration Item. See Get a Configuration Item (content) for details on the object format

## Delete a Configuration Item (content)

**Request**: `DELETE /config-items/{type}/{path}`

**Description**: Deletes a Configuration Item, such as a System Account. Note that this will fail if the item to delete is currently marked as deployed (in other words, its uuid property is not null).

• To delete a deployed Configuration Item, remove it from the Content Pack and redeploy.

• If you want to clear the override value, see Set a Configuration Item's value (content).

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `type` | The type of the Configuration Item to return. For details, see Get a Configuration Item (content). |
| `path` | The relative path of the Configuration Item. For details, see Get a Configuration Item (content). |

**Example:**

`DELETE /config-items/system-accounts/folder1/folder2/sa1`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful |
| 403 | Forbidden | The Configuration Item is currently deployed (its uuid property is not null), or the user is not allowed to manage Configuration Items |
| 404 | Not Found | The Configuration Item does not exist, or the given type is not supported. For System Accounts, this can also mean the user does not have privileges to access this particular item. |

**Response body:**

On success, returns the deleted Configuration Item. See Get a Configuration Item (content) for details on the object format.

## Get Content Configuration tree

**Request**: `GET /config-items/tree`

**Description**: Returns the content configuration tree.

**Example**:

`GET /config-items/tree`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Successful. If there are no results, the result will be empty. |

**Response entity body:**

```
[{
"id":"Configuration",
"parentId":null,
"leaf":false,
"path":"Configuration",
"name":"Configuration",
"type":"FOLDER",
"icon":null

},


{
"id":"Configuration/Group Aliases",
"parentId":"Configuration",
"leaf":false,
"path":"Configuration/Group Aliases",
"name":"Group Aliases",
"type":"FOLDER",
"icon":null

},

{
"id":"Configuration/Group Aliases/RAS_Operator_Path.xml",
"parentId":"Configuration/Group Aliases",
"leaf":true,
"path":"Configuration/Group Aliases/RAS_Operator_Path.xml",
"name":"RAS_Operator_Path",
"type":"GROUP_ALIAS",
"icon":null,
"workerGroup":"RAS_Operator_Path",
"deployed":true

}

]
```

## Get Configuration Item Details

**Request**: `GET /config-items/{type}/{path}?details=true`

**Description**: Returns the deployed entity data of a configuration item

**Example**:

`GET /config-items/system-accounts/folder1/folder2/sa1?details=true`

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `type` | The type of the Configuration Item to return. For details, see Get a Configuration Item (content). |
| `path` | The relative path of the Configuration Item. For details, see Get a Configuration Item (content). |

**Request parameters:**

This API requires the parameter "details=true", otherwise it will be treated as Get a Configuration Item (content). There are no other parameters.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested path was found. |
| 403 | Forbidden | The user is not allowed to view Configuration Items |
| 404 | Not Found | The Configuration Item does not exist, or the given type is not supported. For System Accounts, this can also mean the user does not have privileges to access this particular item |

**Response entity body:**

- **on success**: Returns a JSON object with the following format:

```
{
    "id": "2283acc2-95f0-4cc4-8f45-e5053ace7aaf",
    "name": "some-prop",
    "path": "Configuration/System Properties/some-prop.xml",
    "description": "This is just some property",
    "cpName": "cp-props",
    "version": "10.20.01 "

}
```

# Audit

HPE OO now gives you the option to audit events, so that you can track security breaches. Auditing lets you track actions that took place on Central, such as logins, triggering flows, creating schedules, editing configurations, and so on.

## Get Audit Configuration

**Request**: `GET /audit/config`

**Description**: Gets the audit configuration.

Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested contents were returned. |
| 403 | Forbidden | The user attempting to execute this command does not have the View Security Configuration or the Manage Security Configuration permissions. |

Response entity body:

| Attribute | Type | Description | Comment |
|-----------|------|-------------|---------|
| `enabled` | Boolean | True when auditing is enabled. False when auditing is disabled. | |

Example:

```
{"enabled":true}
```

## Update Audit Configuration

**Request**: `PUT /audit/config`

**Description**: Updates the audit configuration.

**Request entity body:**

| Attribute | Type | Description | Required | Default |
|-----------|------|-------------|----------|---------|
| enabled | Boolean | True to enable auditing. False to disable auditing. | Yes | |

**Example:**

`{"enabled":true}`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The audit configuration was updated successfully. |
| 400 | Bad Request | The data in the request body is invalid. |
| 403 | Forbidden | The user attempting to execute this command does not have the Manage Security Configuration permission. |

**Response entity body:**

Returns the updated audit configuration object.

**Example:**

`{"enabled":true}`

## Get Audit Records

**Request**: `GET /audit/records`

**Description**: Returns a page of the audit records that were recorded in reverse chronological order.

**Request parameters:**

| Attribute | Type | Description | Required | Default |
|---|---|---|---|---|
| `sortDescending` | Boolean | Whether to sort by descending order. If false – sort by ascending order. | No | true |
| `pageNum` | Integer | The number of the returned page. This must be a positive (>0) number. | No | 1 |
| `pageSize` | Integer | The number of records in the returned page. This must be a positive (>0) number. | No | 50 |

**Example:**

`GET /audit/records?sortDescending=false&pageNum=2&pageSize=10`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | The requested contents were returned. |
| 400 | Bad Request | If any of the arguments are invalid. |
| 403 | Forbidden | The user attempting to execute this command does not have the View Audit permission. |

**Response entity body:**

An array which contains all of the events which were audited and meet the filtering criteria. Each element in the array represents an audited event.

| Attribute | Type | Description | Comment |
|---|---|---|---|
| `time` | Long | The audit time | |
| `type` | Predefined Type | The type of operation being audited. Possible values: See the <u>Audit Types/Groups</u> table. | |
| `group` | Predefined Type | The group to which the operation being audited belongs. Possible values: See the <u>Audit Types/Groups</u> table. | |
| `subject` | String | The user that performed the operation being audited. If the operation is a general system event, the subject will be the default user name (anonymousUser). | |
| `outcome` | Predefined Type | The outcome of the operation being audited. Possible values: Success, Failure and System Error | |
| `data` | String | The extra audit data specific to the type of operation being audited. | In JSON key/value format |

**Example:**

```
[
  {
      "time":1412312016740,
      "type":"AuditConfigurationChange",
      "group":"AuditManagement",
      "subject":"mydomain\myuser2",
      "outcome":"Success",
      "data":"{"enabled":false}"
  },
  {
      "time":1412312016722,
      "type":"InternalUserDelete",
      "group":"Authentication-Authorization",
      "subject":"mydomain\myuser2",
      "outcome":"Success",
      "data":{"usersNames":["admin"]}"
  }
]
```

**Example:**

## Delete Audit Records

**Request**: `DELETE /audit/records`

**Description**: Delete the audit records according to the request parameters. Returns the number of records that were actually deleted.

**Request parameters:**

| Attribute | Type | Description | Required | Default |
|-----------|------|-------------|----------|---------|
| `timeBefore` | Long | Records of operations that were audited before this time will be purged. In UTC time format. | Yes | |
| `maxAmount` | Integer | The maximum amount of records to delete.<br>**Note**: In some cases the number of records deleted might be slightly larger than the `maxAmount` that was passed. | No | 1000 |

**Example:**

`DELETE /audit/records?timeBefore=1387218013000&maxAmount=3000`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested contents were deleted. |
| 400 | Bad Request | If any of the arguments are invalid. |
| 403 | Forbidden | The user attempting to execute this command does not have the Manage Data Cleanup permission |

**Response entity body:**

• **on success**: Returns the number of audit records that have been deleted.

**Note:** This number might be higher than the `maxAmount` parameter that was passed.

## Audit Types/Groups

| Type | Group |
|------|-------|
| CentralStartup | CentralLifecycle |
| CentralShutdown | CentralLifecycle |
| AuditConfigurationChange | AuditManagement |
| AuthenticationFailure | Authentication-Authorization |
| AuthorizationFailure | Authentication-Authorization |
| LoginAttempt | Authentication-Authorization |
| Logout | Authentication-Authorization |
| AuthenticationConfigurationUpdate | Authentication-Authorization |
| CaptureCredentialsConfigurationUpdate | Authentication-Authorization |
| RoleCreate | Authentication-Authorization |
| RoleUpdate | Authentication-Authorization |
| RolesDelete | Authentication-Authorization |
| RoleSetDefault | Authentication-Authorization |
| LDAPConfigurationCreate | Authentication-Authorization |
| LDAPConfigurationUpdate | Authentication-Authorization |
| LDAPConfigurationsDelete | Authentication-Authorization |
| InternalUserCreate | Authentication-Authorization |
| InternalUserUpdate | Authentication-Authorization |
| InternalUsersDelete | Authentication-Authorization |
| SAMLConfigurationCreate | Authentication-Authorization |
| SAMLConfigurationUpdate | Authentication-Authorization |
| SAMLConfigurationDelete | Authentication-Authorization |
| SSOConfigurationUpdate | Authentication-Authorization |
| PathEntitlementUpdate | Authentication-Authorization |
| RunTriggered | Runs |
| RunStatusChange | Runs |
| ScheduleCreate | Runs |
| ScheduleEdit | Runs |
| SchedulesEnable | Runs |
| SchedulesDisable | Runs |

| Type | Group |
|------|-------|
| SchedulesDelete | Runs |
| DeploymentProcessCreate | ContentDeployment |
| ContentUploadToDeploymentProcess | ContentDeployment |
| ContentRemoveFromDeploymentProcess | ContentDeployment |
| ContentForDeleteAddToDeploymentProcess | ContentDeployment |
| DeploymentProcessStart | ContentDeployment |
| ContentDeployment | ContentDeployment |
| ContentRollback | ContentDeployment |
| ContentDelete | ContentDeployment |
| GroupAliasCreate | ContentConfiguration |
| GroupAliasUpdate | ContentConfiguration |
| GroupAliasesDelete | ContentConfiguration |
| SystemAccountCreate | ContentConfiguration |
| SystemAccountUpdate | ContentConfiguration |
| SystemAccountsDelete | ContentConfiguration |
| ContentConfigurationItemCreate | ContentConfiguration |
| ContentConfigurationItemUpdate | ContentConfiguration |
| ContentConfigurationItemDelete | ContentConfiguration |
| WorkerRegister | TopologyManagement |
| WorkersDelete | TopologyManagement |
| WorkerUpdate | TopologyManagement |
| WorkersUpdate | TopologyManagement |
| WorkersAssignToGroup | TopologyManagement |
| WorkersRemoveFromGroup | TopologyManagement |
| ExternalUrlCreateOrUpdate | TopologyManagement |
| ExternalUrlDelete | TopologyManagement |
| SystemConfigurationCreateOrUpdate | SystemConfiguration |
| SystemConfigurationUpdate | SystemConfiguration |
| SystemConfigurationDelete | SystemConfiguration |

# LDAP Configuration

The LDAP API allows you to configure you organization's LDAP.

This enables users to log in with their organizational credentials and for the administrator to map LDAP groups to OO Roles.

The LDAP API includes a test API to verify configurations are going to be set correctly before saving them.

**Note**: It is recommended to set LDAP configurations when you want to authenticate users and not rely on the internal users feature, which are less secure.

Although with the LDAP API the configurations are set, you should enable the system authentication for them to take place.

In the case both the LDAP configurations and internal users were set, the LDAP settings override the internal user settings, if there is a collision between user IDs.

### API supported with multiple LDAPs

If your organization works with multiple LDAP servers, it is now possible to configure Central to work with all of them. This includes LDAPs with different schemes and from different vendors. For example, you might have an Active Directory (Microsoft LDAP) implementation for one part of the organization and a Sun One (Oracle LDAP) implementation for another part.

## Get LDAP Configuration by ID

**Request**: `GET /authns/ldap-config/{id}`

**Description**: Retrieves an LDAP configuration according to the specified ID.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | LDAP was retrieved successful. |
| 403 | Forbidden | User does not have view/manage security configurations permissions. |
| 404 | Not found | The requested LDAP does not exist. |

**Response entity body:**

| Attribute | Type | Description |
|-----------|------|-------------|
| `id` | String | The ID for the returned LDAP. |
| `type` | Predefined Value | The type of the LDAP. See the LDAP Appendix for more information. |
| `domain` | String | The domain of the LDAP. |
| `addresses` | An array of Address | The address of the LDAP and other addresses for the case of failover. See the LDAP Appendix for more information. |
| `securedChannel` | boolean | True if a secured channel is used (SSL). |
| `enabled` | boolean | Whether the returned LDAP is operational. |
| `privilegedUser` | String | The DN of a user with search capabilities on the provided User & Group DNs. In case the LDAP type is ACTIVE_DIRECTORY the exact user name will be returned (and not a DN). |
| `privilegedUserPassword` | String | Indicates whether a password for the privileged user was provided. "********" appears when a password was set. |
| `groupsDns` | String[] | The DNs on which to apply the groups filter for search. |
| `groupsSearchRecursive` | boolean | Whether groups search should be recursive. |
| `groupsFilter` | String | A search filter to apply on the groups DNs. |
| `usersDns` | String[] | The DNs on which to apply the users filter for search. |
| `usersSearchRecursive` | boolean | Whether users search should be recursive. |
| `userCommonNameAttribute` | String | The attribute of the user which should be used for display purposes. |
| `userEmailAttribute` | String | The attribute of the user which contains the mail address. |
| `groupMembershipAttribute` | String | In ACTIVE_DIRECTORY type, this represents the attribute of the user which contains the groups. In any other type, it is null. |
| `groupNameAttribute` | String | The attribute of the group which contains the group's name used for mapping Roles to Groups. |
| `usersFilter` | String | A search filter to apply on the users DNs. |

| Attribute | Type | Description |
|---|---|---|
| `customGroupsAttributesNames` | String | Attribute names that will be used as groups. Separated by semicolon. |

**Example**

```
{
"id":"72cdc1a7-1005-4800-a412-5e4a9b8f6bec",
"type":"OPEN_DJ",
"domain":"Indigo",
"addresses":[
        {"port":389,"host":"mysite.com"},
        {"port":389,"host":"mysite.com"},
        {"port":3021,"host":"mysite.com"}
],
"securedChannel":false,
"enabled":true,
"privilegedUser":"uid=user,ou=people,dc=HPE,dc=com",
"privilegedUserPassword":"********",
"groupsDns":[
        "ou=products,dc=HPE,dc=com"
], "groupsSearchRecursive":true,
"groupsFilter":"(uniqueMember={0})",
"groupNameAttribute":"cn",
"usersFilter":"(&(objectclass=person)(uid={0}))",
"usersDns":[
        "ou=people,dc=HPE,dc=com"
], "usersSearchRecursive":true,
"userCommonNameAttribute":"cn",
"userEmailAttribute":"mail",
"groupMembershipAttribute":null
"customGroupsAttributesNames":null

}
```

## Create a New LDAP Configuration

**Request**: `POST /authns/ldap-config`

**Description**: Creates an LDAP configuration.

Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Successful (Created) | LDAP was successfully created |
| 400 | Bad Request | Client didn't provide the mandatory field or provided some fields with wrong format. |
| 403 | Forbidden | User does not have manage security configuration permissions. |
| 409 | Conflict | The provided domain already exists. |

Request body:

| Attribute | Type | Description | Required? | Default value |
|-----------|------|-------------|-----------|---------------|
| `type` | Predefined Value | The type of LDAP. See the LDAP Appendix for more information. | No | OTHER |
| `domain` | String | The domain of the LDAP. For Active Directory this must be a real domain that can be bound on. In all other cases any name is suitable. | Yes | |
| `addresses` | An array of Address | The address of the LDAP and other addresses for the case of fail-over. See the LDAP Appendix for more information. | Yes | |
| `securedChannel` | boolean | True if a secured channel should be used (SSL). | No | False |
| `enabled` | boolean | Whether the provided LDAP should be operational. | No | False |
| `privilegedUser` | String | The DN of a user with search capabilities on the provided User and Group DNs. In case the LDAP type is ACTIVE_DIRECTORY the exact user name should be provided (and not a DN). If the LDAP allows anonymous search, this field is not required. | No | None |
| `privilegedUserPassword` | String | The password of the privileged user. | No | None |

| Attribute | Type | Description | Required? | Default value |
|---|---|---|---|---|
| groupsDns | String[] | The DNs on which to apply the groups filter for search. | Required for types other than ACTIVE_ DIRECTORY and ALTERNATE_ GROUP | |
| groupsSearchRecursive | boolean | Whether groups search should be recursive | No | False |
| groupsFilter | String | A search filter to apply on the groups DNs. | Required for types other than ACTIVE_ DIRECTORY and ALTERNATE_ GROUP | |
| usersDns | String[] | The DNs on which to apply the users filter for search. | Yes | |
| usersSearchRecursive | boolean | Whether users search should be recursive | No | False |
| userCommonNameAttribute | String | The attribute of the user which should be used for display purposes. | Yes | |
| userEmailAttribute | String | The attribute of the user which contains the mail address | No | None |
| groupMembershipAttribute | String | In ACTIVE_ DIRECTORY type, this represents the attribute of the user which contains the groups. In any other type, it is null. | Required only for ACTIVE_ DIRECTORY type. | Mandatory for ACTIVE_ DIRETORY |
| groupNameAttribute | String | The attribute of the group which contains the group's name used for mapping Roles to Groups. | Required for all types other than ALTERNATE_ GROUP. | |
| usersFilter | String | A search filter to apply on the users DNs. | Yes | |
| customGroupsAttributesNames | String | Attribute names that will be used as groups. Separated by semicolon. | Required only for ALTERNATE_ GROUP type | |

**Example:**

```
{ "type":"OPEN_DJ",
"domain":"MyDomain",
"addresses":[{"host":"mysite.com","port":389}]
"securedChannel":false,
"enabled":true,
"privilegedUser":"uid=user,ou=people,dc=HPE,dc=com",
"privilegedUserPassword":"1234",
"groupsDns":[
"ou=products,dc=HPE,dc=com"
], "groupsSearchRecursive":true,
"groupsFilter":"(uniqueMember={0})",
"groupNameAttribute":"cn",
"usersFilter":"(&(objectclass=person)(uid={0}))",
"usersDns":["ou=people,dc=HPE,dc=com"],
"usersSearchRecursive":true,
"userCommonNameAttribute":"cn",
}
```

**Response entity body:**

- `LdapConfig` for successful request. See the response entity table in Get LDAP Configurations.

- `RestError` for unsuccessful response. See the RestError table in the Appendix.

## Get All LDAP Configurations

**Request**: `GET /authns/ldap-config`

**Description**: Retrieves all the configured LDAPs.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | LDAPs were retrieved successfully. |
| 403 | Forbidden | User does not have view/manage security configurations permissions. |

**Response entity body:**

An array of LDAP configurations. An empty array if none exists.

| Attribute | Type | Description |
|-----------|------|-------------|
| `id` | String | The ID for the returned LDAP. |
| `type` | Predefined Value | The type of the LDAP. See the LDAP Appendix for more information. |
| `domain` | String | The domain of the LDAP. |
| `addresses` | An array of Address | The address of the LDAP and other addresses for the case of failover. See the LDAP Appendix for more information. |
| `securedChannel` | boolean | True if a secured channel is used (SSL). |
| `enabled` | boolean | Whether the returned LDAP is operational. |
| `privilegedUser` | String | The DN of a user with search capabilities on the provided User & Group DNs. In case the LDAP type is `ACTIVE_DIRECTORY` the exact user name will be returned (and not a DN). |
| `privilegedUserPassword` | String | Indicates whether a password for the privileged user was provided. "*********" is displayed when a password is set. |
| `groupsDns` | String[] | The DNs on which to apply the groups filter for search. |
| `groupsSearchRecursive` | boolean | Whether groups search should be recursive. |
| `groupsFilter` | String | A search filter to apply on the groups DNs. |
| `usersDns` | String[] | The DNs on which to apply the users filter for search. |
| `usersSearchRecursive` | boolean | Whether users search should be recursive. |
| `userCommonNameAttribute` | String | The attribute of the user which should be used for display purposes. |
| `userEmailAttribute` | String | The attribute of the user which contains the mail address. |
| `groupMembershipAttribute` | String | In `ACTIVE_DIRECTORY` type, this represents the attribute of the user which contains the groups. In any other type, it is null. |
| `usersFilter` | String | A search filter to apply on the users DNs. |
| `groupNameAttribute` | String | The attribute of the group which contains the group's name used for mapping Roles to Groups. |

API Guide | HPE Operations Orchestration Version 10.60

| customGroupsAttributesNames | String | Attribute names that will be used as groups. Separated by semicolon. |
|---|---|---|

## Get Default LDAP

**Request**: `GET /authns/ldap-config/default`

**Description**: Retrieve the default LDAP.

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | LDAPs were retrieved successfully. |
| 403 | Forbidden | User does not have read/manage security configuration permissions. |
| 404 | Not Found | A default LDAP does not exist. |

**Response entity body:**

| Attribute | Type | Description |
|---|---|---|
| `id` | String | The ID for the returned LDAP. |
| `type` | Predefined Value | The type of the LDAP. See the LDAP Appendix for more information. |
| `domain` | String | The domain of the LDAP. |
| `addresses` | An array of Address | The address of the LDAP and other addresses for the case of failover. See the LDAP Appendix for more information. |
| `securedChannel` | boolean | True if a secured channel is used (SSL). |
| `enabled` | boolean | Whether the returned LDAP is operational. |
| `privilegedUser` | String | The DN of a user with search capabilities on the provided User & Group DNs. In case the LDAP type is `ACTIVE_DIRECTORY` the exact user name will be returned (and not a DN). |
| `privilegedUserPassword` | String | Indicates whether a password for the privileged user was provided. "*********" is displayed when a password is set. |
| `groupsDns` | String[] | The DNs on which to apply the groups filter for search. |
| `groupsSearchRecursive` | boolean | Whether groups search should be recursive. |
| `groupsFilter` | String | A search filter to apply on the groups DNs. |
| Attribute | Type | Description |
| `usersDns` | String[] | The DNs on which to apply the users filter for search. |
| `usersSearchRecursive` | boolean | Whether users search should be recursive. |
| `userCommonNameAttribute` | String | The attribute of the user which should be used for display purposes. |
| `userEmailAttribute` | String | The attribute of the user which contains the mail address. |

| Attribute | Type | Description |
|---|---|---|
| groupMembershipAttribute | String | In ACTIVE_DIRECTORY type, this represents the attribute of the user which contains the groups. In any other type, it is null. |
| customGroupsAttributesNames | String | Attribute names that will be used as groups. Separated by semicolon. |
| groupNameAttribute | String | The attribute of the group which contains the group's name used for mapping Roles to Groups. |
| usersFilter | String | A search filter to apply on the users DNs. |

## Delete Default LDAP Marking

**Request**: `DELETE /authns/ldap-config/default`

**Description**: Deletes the default LDAP marking.

**Note**: The LDAP is not deleted.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 204 | No Content | No default LDAP marking exists anymore |
| 403 | Forbidden | User does not have manage security configuration permissions. |

## Create or Update a Default LDAP Marking

**Request**: `PUT/authns/ldap-config/default` or `POST/authns/ldap-config/default`

**Description**: Creates or updates default LDAP marking.

Request body:

| Attribute | Type | Description | Required? | Default value |
|-----------|------|-------------|-----------|---------------|
| `id` | String | The ID that represents the LDAP to be marked as default | Yes | N/A |

Example:

```
{

"id":"72cdc1a7-1005-4800-a412-5e4a9b8f6bec"

}
```

Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The default LDAP was set accordingly. There was a default LDAP before. |
| 201 | Created | The default LDAP was set accordingly. There was no default LDAP before. |
| 400 | Bad Request | The user did not provide a mandatory attribute, or provided an invalid value, for example, the given ID does not correspond with an existing LDAP configuration. |
| 403 | Forbidden | User does not have manage security configuration permissions. |
| 404 | Not Found | The provided ID does not relate to some existing LDAP |

Response entity body:

| Attribute | Type | Description |
|-----------|------|-------------|
| `id` | String | The ID for the returned LDAP. |
| `type` | Predefined Value | The type of the LDAP. See the LDAP Appendix for more information. |
| `domain` | String | The domain of the LDAP. |
| `addresses` | An array of Address | The address of the LDAP and other addresses for the case of failover. See the LDAP Appendix for more information. |
| `securedChannel` | boolean | True if a secured channel is used (SSL). |
| `enabled` | boolean | Whether the returned LDAP is operational. |
| `privilegedUser` | String | The DN of a user with search capabilities on the provided User & Group DNs. In case the LDAP type is `ACTIVE_DIRECTORY` the exact user name will be returned (and not a DN). |
| `privilegedUserPassword` | String | Indicates whether a password for the privileged user was provided. Is displayed "********"when a password is set. |
| `groupsDns` | String[] | The DNs on which to apply the groups filter for search. |
| `groupsSearchRecursive` | boolean | Whether groups search should be recursive. |

| Attribute | Type | Description |
|---|---|---|
| groupsFilter | String | A search filter to apply on the groups DNs. |
| usersDns | String[] | The DNs on which to apply the users filter for search. |
| usersSearchRecursive | boolean | Whether users search should be recursive. |
| userCommonNameAttribute | String | The attribute of the user which should be used for display purposes. |
| userEmailAttribute | String | The attribute of the user which contains the mail address. |
| groupMembershipAttribute | String | In ACTIVE_DIRECTORY type, this represents the attribute of the user which contains the groups. In any other type, it is null. |
| groupNameAttribute | String | The attribute of the group which contains the group's name used for mapping Roles to Groups. |
| customGroupsAttributesNames | String | Attribute names that will be used as groups. Separated by semicolon. |
| usersFilter | String | A search filter to apply on the users DNs. |

## Testing LDAP Configurations

There are two APIs for testing LDAP configurations:

- **Request**: POST /authns/ldap-config/{id}/test
  **Description**: Test an existing LDAP configuration.
- **Request**: POST /authns/ldap-config/test
  **Description**: Test an ad hoc LDAP configuration.

**Request body (only for ad hoc API):**

| Attribute | Type | Description | Required? | Default value |
|---|---|---|---|---|
| userName | String | The user to test the provided configurations with. | Yes | |
| userPassword | String | The password of the user. In case the password is not provided, there will be no authentication attempt. | No | None |
| type | Predefined Value | The type of the LDAP. See the LDAP Appendix for more information. | No | OTHER |
| domain | String | The domain of the LDAP. In case of Active Directory this must be a real domain that can be bind on, in all other cases any name would fit. | Yes | |
| addresses | An array of Address | The address of the LDAP and other addresses for the case of fail-over. | Yes | See the LDAP Appendix for more information |
| securedChannel | boolean | True if a secured channel should be used (SSL) | No | False |
| privilegedUser | String | The DN of a user with search capabilities on the provided User & Group DNs. In case the LDAP type is ACTIVE_ DIRECTORY the exact user name should be provided (and not a DN). If the LDAP allows anonymous search, this field is not required. | No | None |
| privilegedUserPassword | String | The password of the privileged user. If the password of the existing LDAP should be used, displayed with asterisk ("*********"). **Note**: If you decided to provide the password itself, the /authns/ldap- config/test is more suitable. See Ad-hoc Flow Execution. | No | None |
| groupsDns | String[] | The DNs on which to apply the groups filter for search. | Yes | |
| groupsSearchRecursive | boolean | Whether groups search should be recursive | No | False |
| groupsFilter | String | A search filter to apply on the groups DNs | Yes | |
| usersDns | String[] | The DNs on which to apply the users filter for search. | Yes | |

| Attribute | Type | Description | Required? | Default value |
|---|---|---|---|---|
| usersSearchRecursive | boolean | Whether users search should be recursive | No | False |
| userCommonNameAttribute | String | The attribute of the user which should be used for display purposes. | Yes | |
| userEmailAttribute | String | The attribute of the user which contains the mail address | No | None |
| groupMembershipAttribute | String | In ACTIVE_ DIRECTORY type, this represents the attribute of the user which contains the groups. In any other type, it is null. | No | Mandatory for ACTIVE_ DIRETORY |
| groupNameAttribute | String | The attribute of the group which contains the group's name used for mapping Roles to Groups. | Required for all types other than ALTERNATE_ GROUP | |
| usersFilter | String | A search filter to apply on the users DNs. | Yes | |
| enabled | boolean | Whether the returned LDAP is operational | No | False |
| customGroupsAttributesNames | String | Attribute names that will be used as groups. Separated by semicolon. | Required only for ALTERNATE_ GROUP type | |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The provided LDAP configurations were legal and a test was performed. |
| 400 | Bad Request | The provided configurations were bad or server encountered some error while performing the test. |
| 403 | Forbidden | User does not have read/manage security configuration permissions. |
| 404 | Not found | The provided LDAP id does not exist. |

**Response body:**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| authenticated | boolean | Whether or not the user was authenticated. | |
| groupsNames | String[] | An array with the names of the groups the user belongs to. | |
| userAttributes | Attribute[] | An array with the attributes of the user. | See the LDAP Appendix for more information. |
| multipleResults | boolean | Whether or not multiple users with the given name were found. | |

- **RestError for Bad Request** (400 code). See the RestError table for more information.

## Get LDAP root Details

**Request**: `POST /authns/ldap-root or PUT /authns/ldap-root`

**Description**: Retrieves LDAP root details.

**Request body:**

| Attribute | Type | Description | Required? | Default value |
|---|---|---|---|---|
| `addresses` | Address[] | An array of addresses. See the LDAP Appendix for more information. | Yes | |
| `securedChannel` | boolean | Whether to establish a secured connection (SSL) | No | False |

**Example:**

```
{"addresses":[
{"host":"mydwbld0006g.isr.HPE.com","port":389},
{"host":"mydwbld0126g.isr.HPE.com","port":3089}
],
"securedChannel":false
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | LDAPs were retrieved successfully |
| 400 | Bad Request | Client provided illegal/incorrect parameters. |

**Successful response body:**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| `rootDns` | String[] | An array of DNs which represents the LDAProots. | |
| `serverAddress` | Address | The address to which OO successfully connected. See the LDAP Appendix for more information. | |
| `vendorName` | String | The name of the vendor of this LDAP | Not all LDAPs support this functionality |
| `vendorVersion` | String | The version of this LDAP | Not all LDAPs support this functionality |

**Example:**

```
{ "rootDns":[
"dc=mercury,dc=com",
"dc=indigo,dc=com"
], "serverAddress":
{"port":389,"host":"mysite.com"},
"vendorName":"ForgeRock AS.",
"vendorVersion":"OpenDJ 2.6.0"
}
```

- **Bad Request** (400 Code) response. See the RestError table for more information.

## Update an Existing LDAP Configuration

**Request**: PUT /authns/ldap-config/{ldapId}

**Description**: Update an existing LDAP configuration

**Request body:**

| Attribute | Type | Description | Required? | Default value |
|-----------|------|-------------|-----------|---------------|
| `type` | Predefined Value | The type of LDAP. See the <u>LDAP Appendix</u> for more information. | Yes | |
| `domain` | String | The domain of the LDAP. For Active Directory this must be a real domain that can be bound on. In all other cases any name is suitable. | Yes | |
| `addresses` | An array of Address | The address of the LDAP and other addresses for the case of fail-over. See the <u>LDAP Appendix</u> for more information. | Yes | |
| `securedChannel` | boolean | True if a secured channel should be used (SSL). | No | False |
| `enabled` | boolean | Whether the provided LDAP should be operational. | No | False |
| `privilegedUser` | String | The DN of a user with search capabilities on the provided User and Group DNs. In case the LDAP type is `ACTIVE_ DIRECTORY` the exact user name should be provided (and not a DN). If the LDAP allows anonymous search, this field is not required. | No | None |
| `privilegedUserPassword` | String | The password of the privileged user. | No | None |
| `groupsDns` | String[] | The DNs on which to apply the groups filter for search. | Yes | |
| `groupsSearchRecursive` | boolean | Whether groups search should be recursive | No | False |
| `groupsFilter` | String | A search filter to apply on the groups DNs | Yes | |
| `usersDns` | String[] | The DNs on which to apply the users filter for search. | Yes | |
| `usersSearchRecursive` | boolean | Whether users search should be recursive | No | False |
| `userCommonNameAttribute` | String | The attribute of the user which should be used for display purposes. | Yes | |
| `userEmailAttribute` | String | The attribute of the user which contains the mail address | No | None |

| groupMembershipAttribute | String | In ACTIVE_ DIRECTORY type, this represents the attribute of the user which contains the groups. In any other type, it is null. | No | Mandatory for ACTIVE_ DIRETORY |
|---|---|---|---|---|

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful | The LDAP configuration was updated successfully. |
| 400 | Bad Request | Client didn't provide the mandatory field or provided some fields with wrong format. |
| 403 | Forbidden | User does not have manage security configuration permissions. |
| 409 | Conflict | The provided domain already exists. |

**Example:**

```
{
"type":"SUN_ONE",
"domain":"MyDomain",
"addresses":[{"host":"mysite.com","port":389}],
"securedChannel":false,
"enabled":true,
"privilegedUser":"uid=user,ou=people,dc=HPE,dc=com",
"privilegedUserPassword":"1234",
"groupsDns":["ou=products,dc=HPE,dc=com"],
"groupsSearchRecursive":true,
"groupsFilter":"(uniqueMember={0})",
"groupNameAttribute":"cn",
"usersFilter":"(&(objectclass=person)(uid={0}))",
"usersDns":["ou=people,dc=HPE,dc=com"],
"usersSearchRecursive":true,
"userCommonNameAttribute":"cn"
}
```

# Workers

Many deployments can benefit from having more than a single Worker in a specific environment. For example, this could be helpful if you are managing a remote data center in which you need Workers to be able to withstand the action execution load, or simply for high availability of the Workers in that data center. In previous versions, a load balancer would have been required to balance the load between two Workers, which Central would know as a single logical Worker. See the Concepts Guide for more information.

## Get All RASes

**Request**: `GET /rases`

**Description**: Retrieves the information of all the RASes in the system.

**Response status codes**:

| Code | Meaning | Returned When |
| --- | --- | --- |
| 200 | OK | Successful |
| 403 | Forbidden | The user does not have the View Topology or Manage Topology permission. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
[
    {
    "upgradeStatus": "READY_FOR_UPGRADE",
    "upgradeDetails": null,
    "upgradeVersion":null,
    "upgradeDetailsForLoaclization":null,
    "worker":{
        "uuid": "1fc88b1f-33f3-4178-a3ee-996f90f36feb",
        "installPath": "C:\\PROJS\\INSTALLATIONS\\10_60\\ras",
        "os": "Windows 8.1",
        "jvm": "1.8.0_45",
        "description": "1fc88b1f-33f3-4178-a3ee-996f90f36feb",
        "dotNetVersion": "4.x",
        "hostName": "somehost",
        "groups":["RAS_Operator_Path"],
        "status": "RECOVERED",
        "executionStatus": "Idle",
        "monitorInfo": null,
        "ipAddress": null,
        "listenPort": 0,
        "protocolType": null,
        "reversedWorker": false,
        "connectedCentralUuid": null,
        "connectedCentralHostname":null,
        "sharedSecret": null,
        "version": "10.60",
        "versionId": "20160300",
```

```
                    "active": false
                    }
            }
        ]
```

## Prepare RAS Upgrade and Start RAS Upgrade

**Request**: `POST /rases/{uuids}/upgrade`

**Description**: Initiates an upgrade action (either prepare upgrade or start upgrade) for the specified RASes.

**Response status codes**:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The action was initiated for all specified RASes. |
| 403 | Forbidden | The user does not have the **View Topology** or **Manage Topology** permission, or this action is not allowed because one or more of the specified RASes are not in a suitable upgrade state. |
| 404 | Not found | One or more of the specified RAS UUIDs do not exist. |

**Body:**

```
{"action": "PREPARE"}
```
OR
```
{"action": "START"}
```

## Upload RAS Upgrade File

**Request**: `POST /ras-upgrade-file`

**Description**: Uploads the RAS upgrade file to Central. The uploaded file can later be used to upgrade RASes via Central. The file will be overwritten if it already exists.

**Request body:**

The RAS upgrade zip file, which must be uploaded according to the **multipart/form-data** standard (RFC 2388).

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | Successful |
| 400 | Bad Request | The request is not a valid **multipart/form-data** upload, or the provided file is invalid |
| 403 | Forbidden | The user does not have the **Manage Topology** permission |

**Response entity body:**

- **On success**: No body.

- **On failure**: If the failure is due to a bad request, the failure reason will be returned in the following format:

```
{
    "localizedMessage":"Invalid upgrade file. Make sure you are uploading a valid RAS upgrade zip.",
    "details":"Required files are missing from the uploaded zip"
}
```

## Get RAS Upgrade File Version

**Request**: GET /ras-upgrade-file/version

**Description**: Returns the version of the RAS upgrade file that has been uploaded to Central (see <u>Upload RAS Upgrade File</u>).

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | Successful |
| 403 | Forbidden | The user does not have the **View Topology** or **Manage Topology** permission |

**Response entity body:**

- **On success**: The version name in plain text, or empty body if no RAS upgrade is currently stored on Central.

**Example:**

```
10.60
```

## Delete RAS upgrade file

**Request**: `DELETE /ras-upgrade-file`

**Description**: Deletes the RAS upgrade file.

**Examples:**

`DELETE /ras-upgrade-file`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | |
| 403 | Forbidden | Access denied user does not have the manage topology permission or when a RAS is consuming the upgrade file |

## Update a Specific Worker

**Request**: PUT /workers/{workerId}

**Description**: Update an existing worker.

**Request path variables:**

| Attribute | Description |
|---|---|
| workerId | The ID of the worker to be updated. |

**Request entity body:**

The body of this request must include a JSON object with the following format:

**JSON for the worker**

```
{
    "groups":[
        "worker_Operator_Path"
    ],
    "active":false
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | Updated the default role successfully. |
| 403 | Forbidden | The user doesn't have Manage Topology permission. |
| 404 | Not found | The requested worker is not found. |

## Get All Workers

**Request**: GET /workers

**Description**: Retrieves all the workers.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested workers were found. |
| 403 | Forbidden | The user doesn't have **View Topology** or **Manage Topology** permissions. |

**Response entity body:**

• **on success:** Returns a JSON object with the following format:

```
[
    {
        "uuid":"a97e30da-179e-4f19-af93-453c33338f53",
        "installPath":"c:/jenkins/workspace/carmel-demo-deployment/oo/central",
        "os":"Windows Server 2008",
        "jvm":"1.7.0_13",
        "description":"a97e30da-179e-4f19-af93-453c33338f53",
        "dotNetVersion":"4.x", "hostName":"VMCNCDEV41.devlab.ad",
        "groups":[
            "worker_Operator_Path"
        ],
        "active":true,
        "status":"RUNNING"
    },
    {
        "uuid":"4440c50e-79d1-45d2-a8dc-94bc42eb9b1f",
        "installPath":"c:\\jenkins\\workspace\\carmel-demo-deployment\\oo\\worker",
        "os":"Windows Server 2008",
        "jvm":"1.7.0_13",
        "description":"4440c50e-79d1-45d2-a8dc-94bc42eb9b1f",
        "dotNetVersion":"4.x", "hostName":"VMCNCDEV41.devlab.ad",
        "groups":[
            "Worker_Operator_Path"
        ],
        "active":false,
        "status":"RUNNING"
    }
]
```

See the following statuses for more information.

## Worker Status

The following are the possible values of the status attribute, which appears in the APIs:

| Status | Description |
|---|---|
| RUNNING | Normal status |
| IN_RECOVERY | The worker is being recovered, but not finished. Once the recovery is finished it will change the status to **RECOVERED** |
| RECOVERED | Once the recovery is finished it will change the status to **RECOVERED**. If this worker startd again and performs a keepalive to the server it will change to **RUNNING**.  If this worker has permanently stopped it will not change and remain **RECOVERED**. |
| FAILED | This is the initial status when the worker is created before keepalive is sent for the first time |

### Worker Status

## Get All Workers Groups

**Request**: `GET /workers-groups`

**Description**: Return a list of Workers groups.

**Response status code:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | |
| 403 | Forbidden | The user does not have View Topology or Manage Topology permission. |

**Response entity body:**

- **on success:** Returns a JSON array of the Workers Groups with the following format:

```
[
    "RAS_Group_1",
    "RAS_Group_2",
    "RAS_Group_3"

]
```

## Assign Workers to a Workers Group

**Request**: `PUT /workers-groups/{name}/workers/{workersUuids}`

**Description**: Assign Workers to a group.

**Request path variables:**

| Attribute | Description |
|---|---|
| name | The name of the Workers group to add |
| WorkersUuids | The workersUuids of the Worker(s) to be added to the group |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | No Content | |
| 400 | Bad Request | |
| 403 | Forbidden | The user doesn't have Manage Topology permission. |
| 404 | Not Found | The requested worker is not found. |

## Remove Workers from a Workers Group

**Request**: `DELETE /workers-groups/{name}/workers/{workersUuids}`

**Description**: Remove Workers from a Workers Group.

**Request path variables:**

| Attribute | Description |
|---|---|
| name | The name of the Workers Group to remove from |
| workersUuids | The uuid of the Worker(s) to remove from the group |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | No Content | |
| 400 | Bad Request | |
| 403 | Forbidden | The user does not have Manage Topology permission. |
| 404 | Not Found | The requested worker is not found or worker group does not exist. |

## Delete a Worker

**Request**: `DELETE /workers/{workersUuid}`

**Description**: Delete workers.

**Request path variables:**

| Attribute | Description |
|---|---|
| `workersUuid` | List of worker uuids |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | OK | No content, workers deleted successfully. |
| 400 | Bad Request | When you try to delete a worker that is still running. |
| 403 | Forbidden | The user does not have Manage Topology permission. |
| 404 | Not Found | The requested worker is not found. |

## Delete a Worker

# Firewall friendly APIs

## Register Reverse RAS

**Request**: `POST /reverse-rases`

**Description**: Connect to and register reverse RAS.

**Request Entity Body:**

JSON object of worker object

| Attribute | Type | Description | Comments |
|---|---|---|---|
| ipAddress | String | Hostname/IP of the reverse RAS | Mandatory |
| listenPort | Integer | The port on which the reverse RAS is listening | Mandatory |
| sharedSecret | String | Phrase used to authenticate the connection to the reverse RAS | Mandatory |
| protocolType | String | Protocol used to connect to the reverse RAS | Mandatory<br>Possible values:<br>ws – regular connection (http initiated)<br>wss – secured connection (https initiated) |

```
{
      "ipAddress":"10.20.30.40",
      "listenPort":8081,
      "sharedSecret":" Admin111",
      "protocolType":"wss"
}
```

**Examples:**

```
POST /reverse-rases
```

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 201 | Created | |
| 400 | Bad Request | |
| 403 | Forbidden | Access denied  user does not have manage topology permission |
| 500 | Internal Server Error | RAS creation failed. For more details see log files |

**Response entity body:**
**On success:** Returns a JSON object of the new worker with its uuid.
```
{
      "uuid":"d10d4382-ae54-49a1-9e90-2cdfeac4f949",
       "ipAddress":"10.20.30.40",
      "listenPort":1234,
      "sharedSecret":"Admin111",
      "protocolType":"wss"
}
```

# Update Reverse RAS

**Request**: `PUT /reverse-rases/{uuid}`

**Description:** Update reverse RAS details.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `uuid` | RAS unique identifier |

**Request Entity Body:**

JSON object of worker object.

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| ipAddress | String | Hostname/IP of reverse RAS | Mandatory |
| listenPort | Integer | Port reverse RAS listen on | Mandatory |
| sharedSecret | String | Phrase used to authenticate connection to reverse RAS | Mandatory |
| protocolType | String | Protocol used to connect to reverse RAS | Mandatory<br>Possible values:<br>ws – regular connection (http initiated)<br>wss – secured connection (https initiated) |

**Examples:**

```
PUT /reverse-rases/d10d4382-ae54-49a1-9e90-2cdfeac4f949
{
  "ipAddress":"16.60.202.145",
  "listenPort":8081,
  "sharedSecret":"Abcabc12",
  "protocolType":"ws"
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | RAS was updated |
| 400 | Bad Request | Illegal arguments |
| 403 | Forbidden | Access denied user does not have manage topology permission |
| 404 | Not Found | RAS with given uuid doesn't exist |
| 500 | Internal Server Error | RAS update failed. For more details see log files |

**Response entity body:**

**On success**: Returns a JSON object of the updated RAS

# Test Connection to Reverse RAS

**Request**: POST /reverse-rases/test

**Description:**
Open connection to the worker and close it after success.

**Request Entity Body:**

| Attribute | Type | Description | Comments |
|---|---|---|---|
| ipAddress | String | Hostname/IP of reverse RAS | Mandatory |
| listenPort | Integer | Port reverse RAS listen on | Mandatory |
| sharedSecret | String | Phrase used to authenticate connection to reverse RAS | Mandatory |
| protocolType | String | Protocol used to connect to reverse RAS | Mandatory<br>Possible values:<br>ws – regular connection (http initiated)<br>wss – secured connection (https initiated) |

JSON object of worker object

```
{
  "ipAddress":"10.20.30.40",
  "listenPort":1234,
  "sharedSecret":"Admin111",
  "protocolType":"wss"
}
```

**Examples:**

POST /reverse-rases/test

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | Connection tested, regardless of success or failure |
| 400 | Bad Request | Illegal arguments |
| 500 | Internal Server Error | RAS test failed. For more details see log files |

**Response entity body:**

**On success**: Returns test connection status code:
- 0 – test connection success
- 1 – failure, see logs (unable to identify specific error)
- 2 – timeout
- 3 – connection already opened to the requested destination (ip:port)
- 4 – Invalid configuration
- 5 – SSL certificate validation failure
- 6 – Wrong RAS connection (occurs when for already registered RAS we test /edit connection with providing ip:port of different RAS)
- 7 – Invalid credentials (in our case wrong shared secret)
- 8 – No Credentials provided (can be only theoretically)
- 9 – Connection refused (port is closed)

# Get Reverse RAS Proxy Configuration

**Request**: `GET /reverse-ras-proxy`

**Description:** Retrieve proxy configuration for reverse RAS connection.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | Successful (empty body if not defined) |
| 403 | Forbidden | Access denied user does not have manage topology permission |

**Response entity body:**

**on success**: Returns a JSON object with proxy configuration

```
{

        "proxyAddress":"100.201.130.5",
        "proxyListenPort":3322,
        "proxyUserName":"admin",
        "proxyPassword":"admin"


}
```

# Create Reverse RAS Proxy Configuration

**Request**: `POST /reverse-ras-proxy`

**Description**: Create a reverse RAS proxy configuration.

| Attribute | Type | Description | Comments |
|---|---|---|---|
| proxyAddress | String | Hostname/IP of forward RAS | Mandatory |
| proxyListenPort | Integer | Port forward proxy listen on | Mandatory |
| proxyUserName | String | Forward proxy user | Optional |
| proxyPassword | String | Forward proxy password | Optional<br>Required if proxy user provided |

**Examples:**

```
POST /reverse-ras-proxy
```

**Request Entity Body:**

JSON object of proxy configuration object

```
{
        "proxyAddress":"100.201.130.4",
        "proxyListenPort":3322,
        "proxyUserName":"admin",
        "proxyPassword":"admin"
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | Successful |
| 400 | Bad Request | Illegal arguments |
| 403 | Forbidden | Access denied user does not have manage topology permission |

**Response entity body:**

**on success**: Returns a JSON object of the new proxy configuration

# Update Reverse RAS Proxy Configuration

**Request**: `PUT /reverse-ras-proxy`

**Description**: Update reverse RAS proxy configuration.

| Attribute | Type | Description | Comments |
|---|---|---|---|
| proxyAddress | String | Hostname/IP of forward RAS | Mandatory |
| proxyListenPort | Integer | Port forward proxy listen on | Mandatory |
| proxyUserName | String | Forward proxy user | Optional |
| proxyPassword | String | Forward proxy password | Optional<br>Required if proxy user provided |

**Examples:**

**Request Entity Body:**

JSON object of proxy configuration object
```
{
        "proxyAddress":"100.201.130.5",
        "proxyListenPort":3322,
        "proxyUserName":"admin",
        "proxyPassword":"admin"
}
```

**Examples:**

`PUT /reverse-ras-proxy`

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | Successful |
| 400 | Bad Request | Illegal arguments |
| 403 | Forbidden | Access denied user does not have manage topology permission |

**Response entity body:**

**on success**: Returns a JSON object with proxy configuration

# Delete Reverse RAS Proxy Configuration

**Request**: `DELETE /reverse-ras-proxy`

**Description**:

Delete proxy configuration for reverse RAS connection.

**Examples:**

DELETE /reverse-ras-proxy

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | |
| 403 | Forbidden | Access denied user does not have manage topology permission |

**Response entity body:**

**on success**:  No return value

# Users

The Users API allows you to retrieve, update, create and delete users.

## Create New Internal User

**Request**: `POST /users`

**Description**: Adds a new internal user.

**Request entity body:**

The body of this request must include a JSON object with the following format:

**JSON for a user with a password and roles**

```
{
    "username":"mranderson",
    "password":"12345",
    "roles":[
        {"name":"EVERYBODY"},
        {"name":"PROMOTER"}
    ]
}
```

If roles are provided with an empty array, the user is granted with the role that was set as the default.

**Note**: Do not use the `me` user name as this is reserved.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Successful (Created) | An internal user was created successfully. |
| 400 | Bad Request | When the wrong parameters are entered. |
| 403 | Forbidden | The user who executed this command does not have the Manage Security Configuration permission. |
| 409 | Conflict | The given username already exists. |

**Response entity body:**

• **on success:** Returns a JSON object of the created Internal User with the following format:

```
{
    "displayName":"mranderson",
    "userId":"mranderson",
    "emails":null,
    "roles":[
        "END_USER",
        "PROMOTER",
        "SYSTEM_ADMIN"
    ],
    "permissions":null
}
```

## Update Existing User

**Request**: `PUT /users/{username}`

**Description**: Update an existing internal user

**Request path variables:**

| Attribute | Description |
|---|---|
| username | The name of the internal user to update. |

**Request entity body:**

The body of this request must include a JSON object with the following format:

**JSON for an Internal User update with both optional and mandatory fields**:

```
{
    "password": "12345",
    "roles":[
        {"name":"EVERYONE"},
        {"name":"PROMOTER"}
        ],
        "username": "mranderson"
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | Successful (OK) | Updated the user successfully. |
| 400 | Bad Request | |
| 403 | Forbidden | The user who executed this command does not have the Manage Security Configuration permission. |
| 404 | Not Found | The user to update does not exist. |
| 409 | Conflict | Trying to rename a user to a name that already exists. |

**Response entity body:**

- **on success:** Returns a JSON object of the updated internal user.

```
{
    "displayName":"mranderson",
    "userId":"mranderson",
    "emails":null,
    "roles":[
        "END_USER",
        "PROMOTER",
        "SYSTEM_ADMIN"
    ],
    "permissions":null
}
```

## Delete an Internal User

**Request**: `DELETE /users/{userNames}`

**Description**: Deletes users according to a specific list of user names.

**Request path variables:**

| Attribute | Description | Required |
|-----------|-------------|----------|
| `userNames` | The names of internal users to delete. | Yes |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The request was finish with success. |
| 403 | Forbidden | The user who executed this command does not have the Manage Security Configuration permission. |

**Note**: A logged in user cannot delete their own internal user account. In this case the response status will be 200 OK and in the response body the result will be `FORBIDDEN`.

**Response Body:**

A map containing the user Ids as keys and the delete result as values. The values are predefined values: `SUCCESS`, `FORBIDDEN`, `NOT_FOUND`.

**Example:**

```
{
  userName1: "FORBIDDEN",
  userName2: "SUCCESS",
  userName3: "NOT_FOUND"
}
```

## Get Users

Request: `GET /users?domain=internal`

**Description**: Retrieves users

Request parameters:

| Attribute | Description | Default Value | Required |
|-----------|-------------|---------------|----------|
| domain | The location from which the user's provider should be retrieved. Internal stands for internal users. | No | Yes |

Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Returned the requested users list. |
| 403 | Forbidden | The user does not have View Security Configuration or Manage Security Configuration permission. |

Response entity body:

• **on success:** Returns a JSON object with the following format:

```
[
    {
        "displayName":"admin",
        "userId":"admin",
        "emails":null,
        "roles":[
            "ADMIN"
        ],
        "permissions":null
    },
    {
        "displayName":"mranderson",
        "userId":"mranderson",
        "emails":null,
        "roles":[
            "END_USER"
        ],
        "permissions":null
    },

    {
        "displayName":"rothjohn",
        "userId":"rothjohn",
        "emails":null,
        "roles":[
            "EVERYONE"
        ],
        "permissions":null
    }
]
```

## Get Session's User

**Request**: `GET /users/me`

**Description**: Retrieves this session's user.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The session's user was returned. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
{
    "displayName":"admin",
    "userId":"admin",
    "emails":null,
    "roles":[
        "ADMIN"
    ],
    "permissions":[
        "cpManage", "cpRead",
        "topologyManage",
        "flowPermissionManage",
        "topologyRead",
        "securityConfigManage",
        "securityConfigRead",
        "systemSettingsRead",
        "systemSettingsManage",
        "scheduleManage",
        "scheduleRead",
        "configurationItemManage",
        "configurationItemRead",
        "othersRunsManage"
    ]
}
```

# LW SSO

The LW SSO API allows you to configure LW SSO.

## Get LW SSO Configuration

**Request**: `GET /authns/lwsso-config`

**Description**: Retrieves the lightweight SSO configuration.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The LWSSO configuration was returned. |
| 403 | Forbidden | The user does not have View Security Configuration or Manage Security Configuration permission. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
{ "enabled":false,
   "initString":"************",
   "domain":"mydomainnew1.com", "protectedDomains":
          [
             "mydomainnew1.com",
             "mydomainnew2.com"
          ]
}
```

## Update LW SSO configuration

**Request**: PUT /authns/lwsso-config

**Description**: Updates the lightweight SSO configuration.

**Request entity body:**

The body of this request must include a JSON object with the following format (some properties are optional):

```
{
    "enabled":false,
    "initString":"myInitString",
    "domain":"mydomainnew1.com",
    "protectedDomains":
    [
            "mydomainnew1.com",
            "mydomainnew2.com"
    ]
}
```

**Note:**  The **initString** must be at least 12 characters long.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Updated the LW SSO configurations successfully. |
| 400 | Bad Request | |
| 403 | Forbidden | The user does not have Manage Security Configuration permission. |

**Response entity body:**

• **on success:** Returns a JSON object of the saved configurations.

```
{
    "enabled":false,
    "initString":"************",
    "domain":"mydomainnew1.com",
    "protectedDomains":
    [
            "mydomainnew1.com",
            "mydomainnew2.com"
    ]
}
```

# Authentication

The Authentication API allows to enable and disable user authentication.

## Get Authentication Configurations

**Request**: `GET /authns`

**Description**: Retrieves the authentication status

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The authentication status we returned |

**Response entity body:**

```
{
      "enable":true,
      "domains":[
      "Internal"
      ]
}
```

## Update Authentication Configurations

**Request**: PUT /authns

**Description**: Updates the authentication configurations.

**Request body:**

```
{
"enable":true
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 204 | Successful (no-content) | The authentication configuration were updated. |
| 403 | Forbidden | The user does not have Manage Security Configuration permission. |
| 404 | Not Found | The specified path was not found. |

# Roles

The Roles API allows you to configure roles.

## Get Specified Role

**Request**: `GET /roles/{roleName}`

**Description**: Retrieves a role according to the specified role name.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| roleName | The name of the required role. |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested role was found. |
| 403 | Forbidden | The user does not have View Security Configuration or Manage Security Configuration permission. |
| 404 | Not Found | The requested role was not found. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
{
    "name":"ADMINISTRATOR",
    "description":"Administration Role",
    "permissions":[
            "flowPermissionManage",
            "cpManage",
            "dashboardRead",
            "configurationItemRead",
            "cpRead",
            "configurationItemManage",
            "scheduleManage",
            "systemSettingsManage",
            "scheduleRead",
            "securityConfigRead",
            "topologyRead",
            "flowDebug",
            "securityConfigManage",
            "topologyManage",
            "systemSettingsRead",
            "othersRunsManage"
            ],
    "groupsNames":[
        ]
}
```

`groupsNames` attribute refers to the LDAP groups mapping. An empty array indicates that there is no mapping to any LDAP group.

## Get All Roles

**Request**: `GET /roles`

**Description**: Retrieves all the existing roles.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested roles were found. |
| 403 | Forbidden | The user doesn't have View Security Configuration or Manage Security Configuration permission. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

```
[
    {
        "name":"ADMINISTRATOR",
        "permissions":[
            "securityConfigRead",
            "cpRead",
            "topologyManage",
            "securityConfigManage",
            "configurationItemRead",
            "scheduleManage",
            "topologyRead",
            "othersRunsManage",
            "configurationItemManage",
            "systemSettingsManage",
            "flowPermissionManage",
            "cpManage",
            "scheduleRead",
            "systemSettingsRead"
            ],
        "groupsNames":[],
        "description":"Administration Role"
    },
    {
        "name":"EVERYBODY",

"permissions":[],
```

```
        "groupsNames":[],
        "description":"Everybody Role"
    },
    {
        "name":"PROMOTER",
        "permissions":[
            "configurationItemManage",
            "cpRead",
            "configurationItemRead",
            "flowPermissionManage",
            "cpManage"
            ],
         "groupsNames":[],
        "description":"Promoter Role"
    },
    {
        "name":"SYSTEM_ADMIN",
        "permissions":[
            "securityConfigRead",
            "topologyRead",
            "systemSettingsManage",
            "topologyManage",
            "securityConfigManage",
            "systemSettingsRead"
        ],
        "groupsNames":[],
        "description":"System Administrator Role"
    },
    {
        "name":"END_USER",
        "permissions":[],
        "groupsNames":[],
        "description":"End User Role"
    }
]
```

## Create New Role

**Request**: `POST /roles`

**Description**: Adds a new role Request entity body:

The body of this request must include a JSON object with the following format:

**JSON for a role configuration with both optional and mandatory fields:**

```
{ "name":"NEW_ROLE",
  "permissions":[ "cpRead",
          "cpManage"
   ],
   "groupsNames":[],
   "description":"New Role"
}
```

`description` and `groupsNames` are optional.

The `groupsNames` refers to the LDAP groups that should be mapped to this role.

### Response status codes:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Successful (Created) | A new role was created. |
| 400 | Bad Request | The data in the body is incorrect. |
| 403 | Forbidden | The user does not have Manage Security Configuration permission. |
| 409 | Conflict | When the user tries to add a role which is already exists. |

### Response entity body:

• **on success:** Returns a JSON object of the created role with the following format:

```
{
    "name":"Super Power",
    "description":"An all permissions role!",
     "permissions":[
        "flowPermissionManage",
        "cpManage",
        "dashboardRead",
        "configurationItemRead",
        "cpRead",
        "configurationItemManage",
        "scheduleManage",
        "systemSettingsManage",
        "scheduleRead",
        "securityConfigRead",
        "topologyRead", "flowDebug",
        "securityConfigManage",
        "topologyManage",
        "systemSettingsRead",
        "othersRunsManage"
                ],
    "groupsNames":[
        "Super Group"
    ]
}
```

## Update an Existing Role

**Request**: `PUT /roles/{roleName}`

**Description**: Update an existing role

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `roleName` | The name of the role to update. |

**Request entity body:**

The body of this request must include a JSON object with the following format:

JSON for a Role update with both optional and mandatory fields:

```
{
    "description":"Not super power anymore",
    "groupsNames":[
        "Not Super Group"
    ],
    "name":"Not Super Power",
    "permissions":[
        "othersRunsManage",
        "flowPermissionManage",
        "securityConfigRead",
        "securityConfigManage"
    ]
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Updated the role successfully. |
| 400 | Bad Request | The JSON body is incorrect. |
| 403 | Forbidden | The user does not have Manage Security Configuration permission. |
| 404 | Not Found | When the role does not exist. |

**Response entity body:**

• **on success:** Returns a JSON object of the updated internal user.

```
{
    "description":"Not super power anymore", "groupsNames":[
    "Not Super Group"
    ],
    "name":"Not Super
    Power",
    "permissions":[
        "othersRunsManage",
        "flowPermissionManage",
        "securityConfigRead",
        "securityConfigManage"
    ]
}
```

## Delete a Role

**Request**: `DELETE /roles/{roleName}`

**Description**: Deletes a role according to the specified role name.

**Request path variables:**

| Attribute | Description |
|---|---|
| roleName | The identifier of the role name to delete. |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 204 | Successful (no-content) | The role was deleted successfully. |
| 403 | Forbidden | The user does not have Manage Security Configuration permission. |
| 404 | Not found | |

## Delete a Role

## Get the Default Role

**Request**: `GET /roles/default-name`

**Description**: Retrieves the default role.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The default role found. |
| 403 | Forbidden | The user does not have View Security Configuration or Manage Security Configuration permission. |

**Response entity body:**

- **on success:** Returns a JSON object with the following format:

`{"defaultRole":"EVERYBODY"}`

`defaultRole` attribute maps between the default role and an existing one.

## Update the Default Role

**Request**: `PUT /roles/default-name`

**Description**: Update an existing role.

**Request entity body:**

The body of this request must include a JSON object with the following format:

`{"defaultRole":"PROMOTER"}`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Updated the default role successfully. |
| 400 | Bad Request | No such role |
| 403 | Forbidden | The user does not have the **Manage Security Configuration** permission. |

**Response entity body:**

- **on success:** Returns a JSON object of the updated default role.

```
{
  "defaultRole":"PROMOTER"
}
```

## Get Entitlements Per Path and Roles

**Request**: `GET /roles/{rolesNames}/entitlements/{path}`

**Description**: Retrieves the entitlements for the requested roles and path.

**Example:**

`GET /roles/ADMINISTRATOR,EVERYBODY,PROMOTER/entitlements/Library/Flows/flow.xml`

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `rolesNames` | The roles for which the entitlements are requested. |
| `Path` | The full path of the resource, including .xml.<br>**Note**: The "/" in the path must not be encoded. |

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | The requested entitlements were found. |
| 400 | Bad request | |
| 403 | Forbidden | The user does not have **Manage Content Permissions** permission. |
| 404 | Not Found | The requested path was not found. |

**Response entity body:**

• **on success:** Returns a JSON object with the following format:

```
{
    "ADMINISTRATOR":[ "RUN",
        "VIEW"
    ], "EVERYBODY":[
    ], "END_USER":[
    ], "SYSTEM_ADMIN":[
    ], "PROMOTER":[
        "RUN",
        "VIEW"
    ]
}
```

**Note**: Existing entitlements are:

• RUN – run flow,

• VIEW – view flow,

• VIEW_EXECUTE – view and use system account.

## Update Path Entitlement Per Role

**Request**: `PUT /roles/{roleName}/entitlements/{path}`

**Description**: Update the entitlements of the requested role and path. The path can be both a flow and configuration item. The privileges for flows are `RUN` and `VIEW`, and for configuration items, `VIEW_EXECUTE`.

**Example**:

`PUT /roles/SYSTEM_ADMIN/entitlements/Library/cp-parallel`

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| `roleName` | The role for which the entitlements are impact. |
| `Path` | The full path of the resource, including .xml. <br> **Note**: the "/" in the path must not be encoded. |

**Request entity body:**

The body of this request must include a JSON object with the following format:

**JSON for an entitlement path update with both optional and mandatory fields:**

```
{
    "privileges":[
        "VIEW",
        "RUN"
    ],
    "isRecursive":true
}
```

**Note**: The default value for `isRecursive` is `false`.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Updated the role successfully. |
| 400 | Bad Request | The request contains the wrong data, for example, no existing privileges. |
| 403 | Forbidden | The user does not have **Manage Content Permissions** permision. |
| 404 | Not found | The specified path was not found. |

**Response entity body:**

• **on success:** Returns a JSON object of the updated path entitlement on the role.

```
{
    "privileges":[
        "VIEW",
        "RUN"
    ],
    "isRecursive":true
}
```

# System Information and Settings

## Create a System Configuration Item

**Request**: `POST /config`

**Description**: Creates a system configuration item.

**Request entity body:**

The body of this request needs to include a JSON object with the following format:

```
{
        "value":"value",
        "key":"my.test.key"
}
```

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 201 | Created | A system configuration item was created successfully. |
| 403 | Forbidden | The user does not have Manage System Settings permission. |
| 409 | Conflict | The system configuration item is already exists. |

**Response entity body:**

• **on success:** Returns a JSON object of the created system configuration item with the following format:

```
{
        "id": "1179648",
        "key":"myKey",
        "value":"value"
}
```

In addition, a location header containing a URI to retrieve the created system configuration item:

`/config/myKey`

## Get All System Configuration Items

**Request**: `GET /config`

**Description**: Retrieves all system configuration items.

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | All existing system configuration items were retrieved. |
| 403 | Forbidden | The user does not have View System Settings or Manage System Settings permission. |

**Response entity body:**

• **on success:** Returns a JSON object with the following format:

```
{
    "key 1":"value 1",
    "key 2":" value 2",
     ...
    "key n":" value n"

}
```

## Get a System Configuration Item

**Request**: `GET /config/{key}`

**Description**: Retrieves a system configuration item by key.

**Request path variables:**

| Attribute | Description |
|---|---|
| key | The key of the requested system configuration item. |

**Response status codes:**

| Code | Meaning | Returned When |
|---|---|---|
| 200 | OK | The requested system configuration item was retrieved successfully. |
| 403 | Forbidden | The user does not have View System Settings or Manage System Settings permission. |
| 404 | Not Found | No system configuration was retrieved. |

**Response entity body:**

- **on success:** Returns a JSON string with the value of the requested system configuration item.

## Update System Configuration Item

**Request**: PUT /config/{key}

**Description**: Updates an existing system configuration item.

**Request path variables:**

| Attribute | Description |
|-----------|-------------|
| key | The key of the system configuration item to update. |

**Request entity body:**

The body of this request needs to include the new value of the system configuration item, as plain text (no JSON encoding).

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 202 | Accepted | The system configuration item was updated successfully. |
| 403 | Forbidden | The user does not have Manage System Settings permission. |
| 404 | Not Found | The requested system configuration item was not found. |

**Response entity body:**

- **on success:** Returns a JSON value of the updated configuration item's ID.

## Get Database Usage Statistics

**Request:** `GET /db-statistics`

**Description:** Retrieves the HPE OO database usage statistics (in MB) according to the given request parameters.

Note: Database usage represents the sum of database segments/pages in use by HPE OO. It does not reflect the overall "size-on-disk" and/or database transaction log.

**Request parameters:**

| Attribute | Type | Description | Required | Default |
|-----------|------|-------------|----------|---------|
| months | List of numbers | The months to retrieve the statistics for. An empty list means all months. | No | false |
| years | List of numbers | The years to retrieve the statistics for. An empty list means all years. | No | false |

**Example:**

`GET/db-statistics?months=1,2,3&years=2015`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The requested database usage statistics was returned. |
| 403 | Forbidden | The user who executed this command does not have permission for Manage cleanup data or to view or manage system settings. |

**Response entity body:**

An object containing two fields: `lastUpdate`, which represents the last time the database size was checked, and `data`, which is a map containing all of the database statistics that match the request.

The first level of the map is the statistics years. Each year is mapped to the different months, and each month is mapped to the different days. Each day contains the statistics element. This statistics element contains the total usage of the database, the debugger usage, the execution usage and the audit usage for that time (in MB).

**Example:**

```
{
    "lastUpdate":1182139200000,
    "data":
      {
        "2014":{
          "12":{
            "31": {
              "totalUsage": 3,
              "executionUsage": 0,
              "debuggerUsage": 0,
              "auditUsage": 0
            }
          }
        },
        "2015":{
          "1":{
            "1": {
              "totalUsage": 1050,
              "executionUsage": 568,
              "debuggerUsage": 150,
              "auditUsage": 0
            },
            "2": {
              "totalUsage": 3,
              "executionUsage": 756,
              "debuggerUsage": 150,
              "auditUsage": 3
            }
          }
        }
```

```
        }
    }
```

## Get System Log Level

**Request**: `GET /loglevel`

**Description**: Retrieves the system default run log level (persistence level).

**Example**: `GET/loglevel`

**Response status codes**:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The requested log level was returned. |
| 403 | Forbidden | The user does not have the permission View System Settings or Manage System Settings |

**Response entity body:**

- **On success**: Returns a quoted JSON string of the system log level.

**Possible values**: `STANDARD, EXTENDED`

**Example**:

`"STANDARD"`

## Update System Log Level

**Request**: PUT /loglevel

**Description**: Updates the system default run log level (persistence level).

**Example**: PUT /loglevel

**Request entity body**:

The body of this request must include one of the predefined values (quoted JSON string): STANDARD, EXTENDED.

**Example**:

"STANDARD"

**Response status codes**:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 204 | No Content | The requested log level was updated successfully. |
| 400 | Bad Request | The provided log level does not match the predefined values (Case sensitive). |
| 403 | Forbidden | The user does not have the permission Manage System Settings |

## Install License from file

**Request:** `PUT /licensing/file`

**Description:** Install and replace the current license of OO (file).

**Expected header**: Content-Type : multipart/form-data

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The license successfully installed. |
| 400 | Bad Request | The user provided an invalid license file. |
| 403 | Forbidden | User does not have the system manage permission. |

## Install License from key

**Request**: PUT /licensing/key

**Description**: Install and replace the current license of OO (license key).

Expected header: `Content-Type : application/json`

**Response status codes:**

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | OK | The license successfully installed. |
| 400 | Bad Request | The user provided an invalid license file. |
| 403 | Forbidden | User does not have the system manage permission. |

## Install License from key

## Get General Setting by Key

**Request**: `GET /general-settings/{key}`

**Description**: Returns the value of the requested general setting. The following keys are allowed:

- masthead.extra.title
- roi.description
- system.level.flow.timeout

**Example**:

`GET /general-settings/system.level.flow.timeout`

**Response status codes**:

| Code | Meaning | Returned When |
|------|---------|---------------|
| 200 | Successful (OK) | Request is successful and the value is returned. |
| 404 | Not Found | Requested key is not a general settings key and cannot be found. |

# Get HPE OO Version

**Request**: `GET /version`

**Description**: Retrieves information about the HPE OO version.

**Response entity body:**

- **on success**: Returns a JSON object with the HPE OO version information:

```
{
        version: "10.20",
        revision: "61583",
        build ID: "2014-11-02_15-03-32",
        build number: "8",
        build job name: "branch-product
}
```

# Get HPE OO Version

# Appendix

## LDAP

**Address:**

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| host | String | The IP or HOST name of the LDAP | |
| port | int | The port of the LDAP | |

**Attribute:**

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| name | String | The attribute name | |
| value | String | The attribute value | |

**LDAP Type:**

Includes one of the following strings:

- ACTIVE_DIRECTORY
- SUN_ONE
- E_DIRECTORY
- OPEN_DJ
- OPEN_LDAP
- OTHER
- ALTERNATE_GROUP

## General

### RestError

| Attribute | Type | Description | Comments |
|-----------|------|-------------|----------|
| message | String | A localized description of an error that occurred. | |
| Message can be presented to the end user. | | | |