

# Oracle Financial Services

## External Scheduler Interface API Guide



Release 25.08.01

G41443-01

August 2025

ORACLE®

Copyright © 2024, 2025, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

# Contents

## 1 External Scheduler Interface

---

|                                  |    |
|----------------------------------|----|
| Rest API Status Codes            | 1  |
| Execution API                    | 1  |
| Batch Execution API              | 2  |
| Batch Group Execution API        | 3  |
| Execution Status API             | 4  |
| Batch Execution Status API       | 4  |
| Batch Group Execution Status API | 5  |
| Interrupt API                    | 6  |
| Batch Interrupt API              | 7  |
| Batch Group Interrupt API        | 7  |
| Restart API                      | 8  |
| Batch Restart API                | 8  |
| Batch Group Restart API          | 9  |
| Rerun API                        | 10 |
| Batch Rerun API                  | 10 |
| Batch Group Rerun API            | 11 |

## 2 Authenticating for Token Generation

---

|   |   |
|---|---|
| Download the Application Certificate      | 1 |
| Get the OAuth Client ID and Client Secret | 1 |
| Generate the Access Token                 | 2 |
| Generate the Refresh Token                | 3 |
| Invoke the API using the Access Token     | 4 |

# 1

## External Scheduler Interface

External scheduler interface help you to execute the tasks using any Command line utility such as cURL commands. You won't require the application interface to execute the tasks.

You can also integrate the Batches with external schedulers, using the external Scheduler interface.

- [Rest API Status Codes](#)
- [Execution API](#)
- [Execution Status API](#)
- [Interrupt API](#)
- [Restart API](#)
- [Rerun API](#)

## Rest API Status Codes

Refer to the following table for Rest API status codes and their descriptions.

**Table 1-1 Status Codes**

| Status Code | Description   |
|-------------|---|
| 0           | Success   |
| -1          | Failure   |
| -2          | Interrupted   |
| 1           | Not Started   |
| 2           | Ongoing   |
| 3           | Aborted   |
| 4           | Excluded  |
| 5           | Held  |
| -3          | Object does not exist   |
| -4          | Invalid arguments passed in request/not enough params in Request body |
| -5          | Invalid request headers/request headers missing                       |
| -6          | No executable job is present  |
| -7          | Job is already interrupted  |
| -8          | Job is not ongoing/aborted  |

## Execution API

The Execution (POST) API triggers a batch or a batch group.

- **HTTP Method - POST**

- **URL** - /SchedulerService/rest-api/v1/external/trigger
- **Header Parameters**
  - **ofs\_tenant\_id** - Tenant ID of the Application
  - **ofs\_service\_id** - Service ID of the Application
  - **ofs\_workspace\_id** - Workspace ID of the Application. It is defaulted to "WS001" and same should be passed each time.
  - **ofs\_remote\_user** - Used ID of the user. This parameter should be mapped to 'BATCH\_EXEC' function.
  - **locale** - locale in languageCode-countryCode format. For example, en-US.
  - **Authorization: Bearer <token>** - Access token required to authenticate the API. If this token is not provided, 401 Unauthorized error is generated. For more information about Bearer token, refer to [Generate the Access Token](#).
- **Sample cURL Command**

```
curl -i -H "ofs_service_id:<Service ID>" -H "ofs_remote_user:<User ID>" -H
      "ofs_tenant_id:<Tenant ID>" -H "ofs_workspace_id:WS001" -H
      "locale:en-US" -H
      "Content-Type: application/json" -H "Authorization: Bearer
      <BEARER_TOKEN>" -X POST
      <APPLICATION_BASE_PATH>/<URL> -d '<REQUEST_JSON>'
```

## Batch Execution API

Use the Execution API to trigger a batch.

### Attributes

- **batchName** - The unique batch code
- **batchType** - The object type. For Batch, the batch type should be set to `rest`.
- **dynamicParamList** - List of run time parameters which should be overridden over actual values for this trigger. This is an optional parameter.
- **excludedTasks** and **held task** should be comma separated values

### Request Body

```
{
  "batchName": "<BATCH_CODE>",
  "batchType": "rest",
  "excludedTasks": "",
  "heldTasks": "",
  "dynamicParamList": "{ \"batchParams\": { \"FICMISDATE$\": \"<MISDate(yyyy-mm-dd)>\" }, \"taskRuntimeParams\": { \"<TASK CODE1>\": { }, \"<TASK CODE2>\": { } } }"
```

### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
{
  "severity": "info",
  "summary": "Object triggered successfully with Run Id:
batch1_demo_ext_api_2023-12-06_1701839464230_1",
  "batchRunId": "batch1_demo_ext_api_2023-12-06_1701839464230_1",
  "details": "Object triggered successfully.",
  "status": "success",
  "statusCode": "0"
}
```

## Batch Group Execution API

Use the Execution API to trigger a batch group.

- **Attributes**

- `batchName` - The unique batch code.
- `batchType` - The object type. For Batchgroup, the batch type should be set to group.
- `dynamicParamList` - List of run time parameters which should be overridden over actual values for this trigger. This is an optional parameter.
- Exclude and held tasks should be comma separated values of batch code.

### Request Body

```
{
  "batchName": "<BATCHGROUP CODE>",
  "batchType": "group",
  "excludedTasks": "",
  "heldTasks": "",
  "dynamicParamList": "[{\\"batchName\\":\\"<BATCH_CODE1>\\",\\"batchParams\\":
{\\"$FICMISDATE$\\":\\"<MISDate(yyyy-mm-dd)>\\",\\"taskRuntimeParams\\":{\\"<TASK
CODE1>\\":{\\"<TASK CODE2>\\":{\\"}}},
{\\"batchName\\":\\"<BATCH_CODE2>\\",\\"batchParams\\":{\\"$FICMISDATE$
\\":\\"<MISDate(yyyy-mm-dd)>\\",\\"taskRuntimeParams\\":{\\"<TASK_CODE1>\\":
{\\"},\\"<TASK_CODE2>\\":{\\"}}}}]"
}
```

### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
{
  "severity": "info",
  "summary": "Object triggered successfully with Run Id:
batchGroup1_demo_ext_api_2023-12-06_1701840572429_1",
  "batchRunId": "batchGroup1_demo_ext_api_2023-12-06_1701840572429_1",
  "details": "Object triggered successfully.",
  "status": "success",
}
```

```
    "statusCode": "0"
  }
```

## Execution Status API

The Execution Status (POST) API provides the current run status of batch/batch group execution.

- **HTTP Method** - POST
- **URL** - /SchedulerService/rest-api/v1/external/status
- **Header Parameters**
  - **ofs\_tenant\_id** - Tenant ID of the Application.
  - **ofs\_service\_id** - Service ID of the Application.
  - **ofs\_workspace\_id** - Workspace ID of the Application. It is defaulted to "WS001" and same should be passed each time.
  - **ofs\_remote\_user** - Used ID of the user. This parameter should be mapped to 'BATCH\_EXEC' function.
  - **locale** - locale in languageCode-countryCode format. For example, en-US.
  - **Authorization: Bearer <token>** - Access token required to authenticate the API. If this token is not provided, 401 Unauthorized error is generated. For more information about Bearer token, refer to [Generate the Access Token](#).
- **Sample cURL Command**

```
curl -i -H "ofs_service_id:<Service ID>" -H "ofs_remote_user:<User ID>" -H
    "ofs_tenant_id:<Tenant ID>" -H "ofs_workspace_id:WS001" -H
"locale:en-US" -H
    "Content-Type: application/json" -H "Authorization: Bearer
<BEARER_TOKEN>" -X POST
    <APPLICATION_BASE_PATH>/<URL> -d '<REQUEST_JSON>'
```

Topics:

- [Batch Execution Status API](#)
- [Batch Group Execution Status API](#)

## Batch Execution Status API

Use the Execution Status API to view the current run status of a batch execution.

### Attributes

- **batchRunId** - Execution ID generated while triggering the object and can be obtained in the response of Execution API.
- **tasks** - List of Task Codes. This is an optional parameter.

### Request Body

```
{
  "batchRunId": "<Batchrun_ID>",
}
```

### Sample Response Body

The following Response body is a sample for `Success : 200 OK`. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```

{
  "severity": "info",
  "batchRunId": "BatchTEST1_2022-05-31_1653994545003_1",
  "taskStatusList": [
    {
      "taskCode": "t1",
      "taskStatus": "SUCCESSFUL",
      "statusCode": "0"
    },
    {
      "taskCode": "t5",
      "taskStatus": "FAILED",
      "statusCode": "-1"
    }
  ],
  "batchStatusCode": "-1",
  "batchList": [],
  "batchStatus": "FAILED",
  "status": "success",
  "statusCode": "0"
}

```

## Batch Group Execution Status API

Use the Execution Status API to view the current run status of a batch group execution.

Use the Execution Status API to view the current run status of a batch group execution.

### Attributes

- `batchRunId` - Execution Id generated while triggering the object and can be obtained in the response of Execution API.

### Request Body

```

{
  "batchRunId": "<Batchrun_ID>"
}

```

### Sample Response Body

The following Response body is a sample for `Success : 200 OK`. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```

{
  "severity": "info",
  "batchRunId": "AbTestBG001_2023-01-27_1674798339245_1",
  "batchStatusCode": "0",
  "batchList": [
    {

```



```

        "batchRunId": "AbTestBatch002_2023-01-27_1674798339462_1",
        "batchStatusCode": "0",
        "batchStatus": "SUCCESSFUL"
    },
    {
        "batchRunId": "AbTestBatch003_2023-01-27_1674798339556_1",
        "batchStatusCode": "0",
        "batchStatus": "SUCCESSFUL"
    }
],
"batchStatus": "SUCCESSFUL",
"status": "success",
"statusCode": "0"
}

```

## Interrupt API

The Execution Status (POST) API Interrupts a batch/batch group execution.

- **HTTP Method** - POST
- **URL** - /SchedulerService/rest-api/v1/external/interrupt
- **Header Parameters**
  - **ofs\_tenant\_id** - Tenant ID of the Application.
  - **ofs\_service\_id** - Service ID of the Application.
  - **ofs\_workspace\_id** - Workspace ID of the Application. It is defaulted to "WS001" and same should be passed each time.
  - **ofs\_remote\_user** - Used ID of the user. This parameter should be mapped to 'BATCH\_EXEC' function.
  - **locale** - locale in languageCode-countryCode format. For example, en-US.
  - **Authorization: Bearer <token>** - Access token required to authenticate the API. If this token is not provided, 401 Unauthorized error is generated. For more information about Bearer token, refer to [Generate the Access Token](#).
- **Sample cURL Command**

```

curl -i -H "ofs_service_id:<Service ID>" -H "ofs_remote_user:<User ID>" -H
    "ofs_tenant_id:<Tenant ID>" -H "ofs_workspace_id:WS001" -H
"locale:en-US" -H
    "Content-Type: application/json" -H "Authorization: Bearer
<BEARER_TOKEN>" -X POST
    <APPLICATION_BASE_PATH>/<URL> -d '<REQUEST_JSON>'

```

### Related Topics

- [Batch Interrupt API](#)  
Use the Interrupt API to interrupt a batch execution.
- [Batch Group Interrupt API](#)  
Use the Interrupt API to interrupt a batch group execution.

## Batch Interrupt API

Use the Interrupt API to interrupt a batch execution.

### Attributes

- `batchName` - The unique batch code
- `batchRunID` - Execution ID generated while triggering the object and can be obtained in the response of Execution API.

### Request Body

```
{
  "batchName": "<Batch_code>",
  "batchRunId": "<Batchrun_ID>"
}
```

### Sample Response Body

The following Response body is a sample for `Success : 200 OK`. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
{
  "severity": "info",
  "batchRunId": "B2001_2022-05-30_1653233511394_1",
  "details": "Execution interrupted successfully.",
  "statusCode": "0",
  "status": "success"
}
```

## Batch Group Interrupt API

Use the Interrupt API to interrupt a batch group execution.

### Attributes

- `batchName` - The unique batch code
- `batchRunID` - Execution ID generated while triggering the object and can be obtained in the response of Execution API.

### Request Body

```
{
  "batchName": "<Batchgroup_code>",
  "batchRunId": "<Batchrun_ID>"
}
```

### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
{
  "severity": "info",
  "batchRunId": "B2001_2022-05-30_1653233511394_1",
  "details": "Execution interrupted successfully.",
  "statusCode": "0",
  "status": "success"
}
```

## Restart API

The Restart (POST) API restarts a batch/batch group execution.

- **HTTP Method** - POST
- **URL** - /SchedulerService/rest-api/v1/external/restart
- **Header Parameters**
  - **ofs\_tenant\_id** - Tenant ID of the Application.
  - **ofs\_service\_id** - Service ID of the Application.
  - **ofs\_workspace\_id** - Workspace ID of the Application. It is defaulted to "WS001" and same should be passed each time.
  - **ofs\_remote\_user** - Used ID of the user. This parameter should be mapped to 'BATCH\_EXEC' function.
  - **locale** - locale in languageCode-countryCode format. For example, en-US.
  - **Authorization: Bearer <token>** - Access token required to authenticate the API. If this token is not provided, 401 Unauthorized error is generated. For more information about Bearer token, refer to [Generate the Access Token](#).
- **Sample cURL Command**

```
curl -i -H "ofs_service_id:<Service ID>" -H "ofs_remote_user:<User ID>" -H
  "ofs_tenant_id:<Tenant ID>" -H "ofs_workspace_id:WS001" -H
"locale:en-US" -H
  "Content-Type: application/json" -H "Authorization: Bearer
<BEARER_TOKEN>" -X POST
  <APPLICATION_BASE_PATH>/<URL> -d '<REQUEST_JSON>'
```

### Related Topics

- [Batch Restart API](#)  
Use the Restart API to restart a batch execution.
- [Batch Group Restart API](#)  
Use the Restart API to restart a batch group execution.

## Batch Restart API

Use the Restart API to restart a batch execution.

### Attributes

- `batchName` - The unique batch code
- `batchRunID` - Execution ID generated while triggering the object and can be obtained in the response of Execution API.

#### Request Body

```
{
  "batchName": "<Batch_code>",
  "batchRunId": "<Batchrun_ID>"
}
```

#### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
{
  "severity": "info",
  "summary": "Object triggered successfully for restart with Run Id: B0001_2022-04-30_1651731208588_1",
  "batchRunId": "B0001_2022-04-30_1651731208588_1",
  "details": "Object triggered successfully.",
  "statusCode": "0",
  "status": "success"
}
```

## Batch Group Restart API

Use the Restart API to restart a batch group execution.

#### Attributes

- `batchName` - The unique batch code
- `batchRunID` - Execution ID generated while triggering the object and can be obtained in the response of Execution API.

#### Request Body

```
{
  "batchName": "<Batchgroup_code>",
  "batchRunId": "<Batchrun_ID>"
}
```

#### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
{
  "severity": "info",
  "summary": "Object triggered successfully for restart with Run Id: B0001_2022-04-30_1651731208588_1",
  "batchRunId": "B0001_2022-04-30_1651731208588_1",
  "details": "Object triggered successfully.",
  "statusCode": "0",
  "status": "success"
}
```

```
"batchRunId": "B0001_2022-04-30_1651731208588_1",  
"details": "Object triggered successfully.",  
"statusCode": "0",  
"status": "success"  
}
```

## Rerun API

The Rerun (POST) API helps to rerun a batch/batch group execution.

- **HTTP Method** - POST
- **URL** - /SchedulerService/rest-api/v1/external/rerun
- **Header Parameters**
  - **ofs\_tenant\_id** - Tenant ID of the Application.
  - **ofs\_service\_id** - Service ID of the Application.
  - **ofs\_workspace\_id** - Workspace ID of the Application. It is defaulted to "WS001" and same should be passed each time.
  - **ofs\_remote\_user** - Used ID of the user. This parameter should be mapped to 'BATCH\_EXEC' function.
  - **locale** - locale in languageCode-countryCode format. For example, en-US.
  - **Authorization: Bearer <token>** - Access token required to authenticate the API. If this token is not provided, 401 Unauthorized error is generated. For more information about Bearer token, refer to [Generate the Access Token](#).
- **Sample cURL Command**

```
curl -i -H "ofs_service_id:<Service ID>" -H "ofs_remote_user:<User ID>" -H  
      "ofs_tenant_id:<Tenant ID>" -H "ofs_workspace_id:WS001" -H  
"locale:en-US" -H  
      "Content-Type: application/json" -H "Authorization: Bearer  
<BEARER_TOKEN>" -X POST  
      <APPLICATION_BASE_PATH>/<URL> -d '<REQUEST_JSON>'
```

### Related Topics

- [Batch Rerun API](#)  
Use the Rerun API to rerun an existing batch execution.
- [Batch Group Rerun API](#)  
Use the Rerun API to rerun an existing batch group execution.

## Batch Rerun API

Use the Rerun API to rerun an existing batch execution.

### Attributes

- **batchName** - The unique batch code
- **batchRunID** - Execution ID generated while triggering the object and can be obtained in the response of Execution API.

### Request Body

```
{
  "batchName": "<Batch_code>",
  "batchRunId": "<Batchrun_ID>"
}
```

### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
Success Scenario: 200 OK
{
  "severity": "info",
  "summary": "Object triggered successfully for rerun with Run Id:
B2001_2022-05-30_1653223084727_1",
  "batchRunId": "B2001_2022-05-30_1653223084727_1",
  "details": "Object triggered successfully.",
  "statusCode": "0",
  "status": "success"
}
```

## Batch Group Rerun API

Use the Rerun API to rerun an existing batch group execution.

### Attributes

- `batchName` - The unique batch code
- `batchRunID` - Execution ID generated while triggering the object and can be obtained in the response of Execution API.

### Request Body

```
{
  "batchName": "<Batchgroup_code>",
  "batchRunId": "<Batchrun_ID>"
}
```

### Sample Response Body

The following Response body is a sample for Success : 200 OK. For more information about status code in the response body, refer to [Rest API Status Codes](#).

```
Success Scenario: 200 OK
{
  "severity": "info",
  "summary": "Object triggered successfully for rerun with Run Id:
B2001_2022-05-30_1653223084727_1",
  "batchRunId": "B2001_2022-05-30_1653223084727_1",
  "details": "Object triggered successfully.",
  "statusCode": "0",
}
```

```
    "status": "success"  
  }
```

## 2

# Authenticating for Token Generation

An Authentication token is required to invoke an API to generate the File Upload/Download PAR URL. The Authentication Process for token generation, utilizes cURL Commands in a CLI Tool to generate the access token and invoke REST APIs.

The Authentication Token is generated through the OAuth Client ID and Secret Credentials created in IAM during Provisioning. The Authentication Token does not require that you log in to the required Cloud Service to invoke the REST APIs from external applications.

Ensure that you have the appropriate log-in credentials to access the required Cloud Service and the appropriate roles to perform specific operations using the API Resources. Below is a list of authentication steps, with subsequent sections offering detailed information:

1. [Download application certificate](#)
2. [Get the OAuth Client ID and Client Secret](#)
3. [Generate the access token](#)
4. [Generate refresh token](#)
5. [Invoke API using the access token](#)

## Download the Application Certificate

The Application Certificate is required for verification purposes when you use cURL commands.

You may choose not to download the certificate if you plan to turn off the cURL Certificate Verification and use an insecure connection (if you add the `--insecure` Flag to the cURL command).

To download the Application Certificate:

1. Log in to your Cloud service.
2. Click **View site information/Verified by** in the Browser URL Address Bar.
3. Select **More information**, to view the certificate.
4. Click **View Certificate** and then click **PEM(cert)** to download the certificate.

## Get the OAuth Client ID and Client Secret

An OAuth Client ID and Client secret are required to generate an access token.

To get the OAuth Client ID and Client Secret:

1. Enter the **Oracle Identity and Access Management (IAM)** URL in the browser's Address bar to access the **Oracle Cloud Account Sign In** page.
2. Log in to **IAM** portal.
3. Click **Navigation** to view a list of available functions.
4. Select **Oracle Cloud Services**.



For more information, see [Access Service Consoles](#) from **Administering Oracle Identity Cloud Service**.

5. From the Oracle Cloud Services, select the required Cloud Service (in **<Cloud\_service\_name> <tenant-id>** format) from the list. For example, the cloud service name - **AMLCS tenant-prd**
6. Click the **Configuration** tab.  
The Client ID and Client Secret Details are displayed in the General Information section.
7. Copy the Client ID and Client Secret.
8. Open a CLI Tool.
9. Proceed to [generate the access token](#).

You can also [get the OAuth client ID and client secret using Admin Console](#).

## Generate the Access Token

Access token is required to invoke API and you can generate an access token using cURL commands.

To generate the Access Token, add the Client ID, Client Secret, User Name, and Password using cURL Commands in the CLI Tool. You can use an insecure connection (if you add the `--insecure / -k` Flag to the cURL command). The following is an example:

```
curl -k -i -H "Authorization: Basic < Base64 Encoded
      Outh Cred >" -H "Content-Type: application/x-www-form-
urlencoded; charset=UTF-8"
      --request POST https://<iam_tenant>:443/oauth2/v1/token -d
```

```
"grant_type=password&scope=urn:opc:idm:__myscopes__+offline_access&username=<u
serid>&password=<Password>"
```

### Sample Code

```
curl -k -i -H "Authorization: Basic
YWFpdGVzdGRldjEwMDEtcHJkX0FQUElEOjQyYjJlYWVlLTl0GEtNDgzYilhMWI2LTBlYzU0MzBmYW
QwNQ==" -H "Content-Type: application/x-www-form-urlencoded; charset=UTF-8" --
request POST https://
iam-0cb0c2b3ba624afca67467fd5eb9db49.identity.c9dev2.oc9qadev.com:443/
oauth2/v1/token -d
"grant_type=password&scope=urn:opc:idm:__myscopes__+offline_access&username=cn
eadmin&password=Password@12345"
```

After generating the Access Token, invoke the API as shown in the following section.

#### Note

The Access token expiry (in seconds) is configurable and can be set at the time of generating the access token. In the preceding example, it is set to 3600 seconds ~ 1 hour. By default, the expiry is set to 3600 seconds ~ 1 hour. You can configure this to a value of your choice up to a maximum value of 31536000 seconds ~ 1 year.

The token is sent as a response. Store the token in a secure location.

#### Sample Access Token (Truncated example)

```
{ "access_token": "eyJ4NXQjUzI1NiI6I1F5azRtb3pIakhuQjJoQnVWdmZXZUpVeVZrNHhUdWd6a
WpHSC1pV21xb1EiLCJ4NXQiOiJDRFhHYVlWZXI3STVhQ1l
...
...
DB_be0RtWlaMxFYg8Ft0VaKl4wOVFGHgg1Cr6GiNvbgeYRG5uWgJGqW", "token_type": "Bearer"
, "expires_in": 3600, "refresh_token": "AgAgYjA1OGVlMjJiMWY2NGU3YWFKM2NjZWNlOTc2Mj
NiNDgIABBMZRHxpaHil2VBXkevFX-
iAAAAMMq9uQDo86eVVVisw3kYn80iX8qRJ2m7hMLmMAhldY9Wgy-ESu8WYzdTBXOsnwHr7A==" }
```

## Generate the Refresh Token

Refresh tokens are used to generate access tokens for invoking APIs.

To generate a Access token using Refresh token, use the following Curl command. You can use an insecure connection (if you add the `--insecure / -k` flag to the cURL command). The following is an example:

```
curl -k -i -H "Authorization: Basic <base64Encoded clientid:secret>" -H
"Content-Type:
    application/x-www-form-urlencoded;charset=UTF-8" --request POST
    https://<IdentityDomainURL>/oauth2/v1/token -d

"scope=urn:opc:idm:__myscopes__&grant_type=refresh_token&refresh_token=<refres
h_token>"
```

#### Sample Code

```
curl -k -i -H "Authorization: Basic

cWppMHBkLXByZF9BUFBjRDplZjFjMTVmZi1lZDBiLTQxNmItYTfmYy0wNjhlYzM5NmUxM2Y=" -H
"Content-Type: application/x-www-form-urlencoded;charset=UTF-8" --
request POST
    https://<IdentityDomainURL>/oauth2/v1/token -d

"scope=urn:opc:idm:__myscopes__&grant_type=refresh_token&refresh_token=AgAgYjA
1OGVlMjJiMWY2NGU3YWFKM2NjZWNlOTc2MjNiNDgIABBMZRHxpaHil2VBXkevFX-
iAAAAMMq9uQDo86eVVVisw3kYn80iX8qRJ2m7hMLmMAhldY9Wgy-ESu8WYzdTBXOsnwHr7A=="
```

#### Sample Refresh Token (Truncated example)

```
{ "access_token": "eyJ4NXQjUzI1NiI6I1F5azRtb3pIakhuQjJoQnVWdmZXZUpVeVZrNHhUdWd6a
WpHSC1pV21xb1EiLCJ4NXQiOiJDRFhHYVlWZXI3STVhQ1l...
...
...
token_type": "Bearer", "expires_in": 3600, "refresh_token": "AgAgYjA1OGVlMjJiMWY2NG
U3YWFKM2NjZWNlOTc2MjNiNDgIABA4t8V_dYVyc5l0uKezofTUAAMJrpmKRhDWf3-
ejCreU8_Po5Bb95srwUDDs5cVlgT-x26twbAfp_ffMCiEgjqGeDNw==" }
```

## Invoke the API using the Access Token

After creating an access token using OAuth Client ID and Client secret, you can invoke the Specific API.

To invoke the API using the generated Access Token, refer to the following example executed using cURL Commands in the CLI Tool:

```
curl -iL -H "Authorization: Bearer <access token>" -H "Content-Type:
<content_type>" -d "<request_body>" --cacert <certificate(.pem)> -X
<http_verb> <api_url>

curl -iL -H "Authorization: Bearer <AUTH_TOKEN>"

-H "Content-Type: application/json" -d "{\"type\":\"files\",\"data\":
[{\"fileName\":\"testtoken\",\"mimeType\":\"text/plain\",\"fileSize\":
123}]]}" --cacert outcert.pem -X POST https://<OCI-URL>/<TENANT><APP_ID>/dsa/
utils/getObjStoreParUrl
```