

LakeFormation

Development Guide

Issue 01
Date 2025-02-14



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2025. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 About This Document.....	1
1.1 Application Development.....	1
1.2 Development Process.....	1
2 Environment Preparation.....	2
2.1 Overview.....	2
2.2 Preparing a LakeFormation Instance.....	2
2.3 Creating a Client.....	3
2.4 Modifying DNS Information.....	3
3 Using LakeFormation Java SDK to Develop Programs.....	5
3.1 Preparing the Development Environment.....	5
3.2 Reference Example.....	11
3.3 Commissioning Applications.....	12

1 About This Document

1.1 Application Development

DataArts Lake Formation (LakeFormation) is a one-stop enterprise-level data lake building service.

It allows you to manage data lake metadata through a graphical user interface (GUI) and APIs, utilizing a decoupled compute and storage architecture. It is compatible with the Hive metadata model and Ranger permission model, and it connects seamlessly to various compute engines and big data services.

This document describes the application development process, environment preparation, and program compilation of LakeFormation.

1.2 Development Process

The LakeFormation development process is as follows:

Table 1-1 Development process description

Step	Description	Reference
Preparing the environment	Prepare the development environment.	Environment Preparation
Writing programs based on scenarios	Use LakeFormation SDK to write programs	Using LakeFormation Java SDK to Develop Programs

2 Environment Preparation

2.1 Overview

Table 1 describes the environment required for developing lake warehouse building applications.

You also need to prepare an environment for verifying whether an application is running properly.

Table 2-1 Preparation items


Item	Description
Preparing a LakeFormation instance	Create an instance on the LakeFormation console or use an existing instance.
Creating a client	Access the LakeFormation console and create a client on the Clients page.
Modifying DNS information	Access the DNS console and modify the private domain name of the subnet where the Linux environment is located.

2.2 Preparing a LakeFormation Instance

- Complete account registration and authorization if you use LakeFormation for the first time. For details, see [Preparations](#).
- For how to create a LakeFormation instance, see [Creating an Instance](#).
- The AK and SK of the target Huawei Cloud account has been created. For details, see section [Obtaining AK/SK](#).

2.3 Creating a Client

Step 1 Log in to the Huawei Cloud management console.

Step 2 In the upper left corner, click  and choose **Analytics > LakeFormation** to access the LakeFormation console.

Step 3 Select the target LakeFormation instance from the drop-down list on the left to access the instance page.

Step 4 Click **Clients** in the navigation pane.

Step 5 Click **Create**. In the displayed dialog box, set the following parameters and click **OK**.

If no suitable VPC or subnet is available, click **create one** to access the VPC console to create one.

Table 2-2 Parameters for creating a client

Parameter	Description
Client	LakeFormation client name.
VPC	VPC where the running and commissioning environment is located.
Subnet	Subnet where the environment for running and commissioning is located.

Step 6 After the access client is created, return to the **Clients** page to view the information about the newly created access client. Wait until the client is created. It is created when its status changes to **Running**.

Step 7 Click **View Details** to view the access client information and record the access IP address.

----End

2.4 Modifying DNS Information

Step 1 Log in to the Huawei Cloud management console.

Step 2 Click the service list icon and choose **Networking > Domain Name Service**.

Step 3 In the navigation pane, choose **Private Zones**.

Step 4 Search for **lakeformation.lakecat.com** in the search box, locate the row that contains the VPC domain name corresponding to the created client in the **Associated VPC** column, and click **Manage Record Set** in the **Operation** column.

Step 5 Click **Add Record Set**, set the following parameters, and click **OK**.

Table 2-3 Parameters for adding a record set

Parameter	Description
Type	Select A - Map domains to IPv4 addresses .
TTL (s)	Cache duration of the record set on a local DNS server. Set this parameter as required.
Value	Enter the client access IP address, which can be obtained by performing the operation described in Step 7 .

NOTICE

When you add a record set, you might see a message that says it conflicts with an existing one. If this happens, look for any conflicting record sets in the existing one.

Step 6 After a record set is added, you can find it in the record set list.

----End

3 Using LakeFormation Java SDK to Develop Programs

3.1 Preparing the Development Environment

Preparing the Environment

Table 3-1 describes the preparations required before using LakeFormation Java SDK.

Table 3-1 Environment requirements

Item	Description
Java JDK environment	The Java environment is required and the Java version must be JDK 1.8 or later.
IntelliJ IDEA	<p>Tool used for developing applications. The version must be 2019.1 or other compatible versions.</p> <p>NOTE</p> <ul style="list-style-type: none"> • If you are using an IBM JDK, ensure that the JDK configured in IntelliJ IDEA is the IBM JDK. • If you are using an Oracle JDK, ensure that the JDK configured in IntelliJ IDEA is the Oracle JDK. • If you are using an open JDK, ensure that the JDK configured in IntelliJ IDEA is the Open JDK. • Do not use the same workspace and the sample project in the same path for different IntelliJ IDEA projects.
Maven installation	Basic configuration of the development environment. This tool is used for project management throughout the lifecycle of software development.
7-Zip	This tool is used to decompress .zip and .rar packages. The 7-Zip 16.04 version is supported.

Collecting Dependency Information

- **Collecting LakeFormation Java SDK dependencies**

View the JAR package of LakeFormation Java SDK of the latest version in the Maven repository at [Maven SDK address](#) and obtain the file content. The following is an example.

```
<dependency>
<groupId>com.huaweicloud.sdk</groupId>
<artifactId>huaweicloud-sdk-lakeformation</artifactId>
<version>3.1.45</version>
</dependency>
```

- **Preparing maven-assembly-plugin dependencies**

Prepare the following maven-assembly-plugin dependencies in advance:

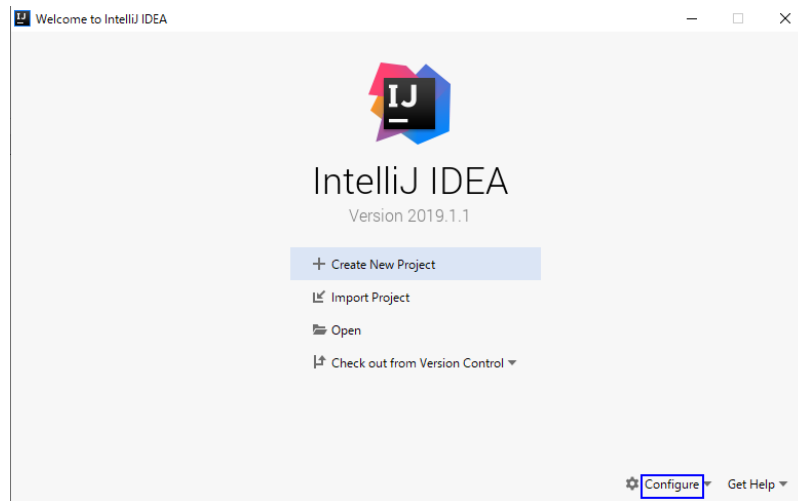
```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-assembly-plugin</artifactId>
      <version>3.3.0</version>
      <configuration>
        <descriptorRefs>
          <descriptorRef>jar-with-dependencies</descriptorRef>
        </descriptorRefs>
        <archive>
          <manifest>
            <mainClass>com.huawei.cloud.dalf.lakecat.examples.CatalogExample</mainClass>
          </manifest>
        </archive>
      </configuration>
    </plugin>
  </plugins>
</build>
```

Configuring IntelliJ IDEA and POM Files

Step 1 After the IntelliJ IDEA and JDK tools are installed, configure the JDK in IntelliJ IDEA.

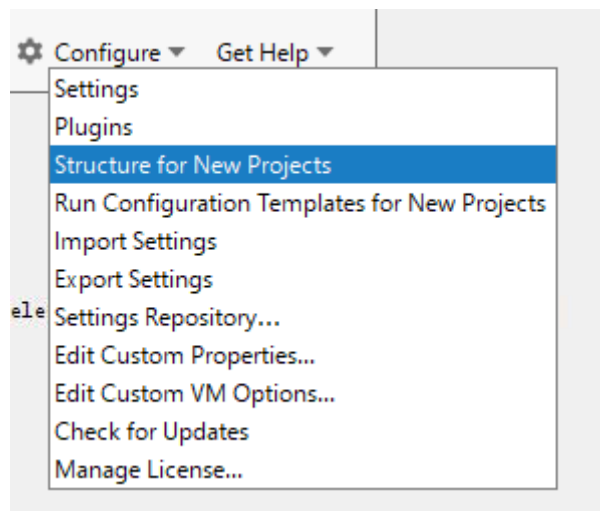
1. Start IntelliJ IDEA and choose **Configure**.

Figure 3-1 Quick Start page



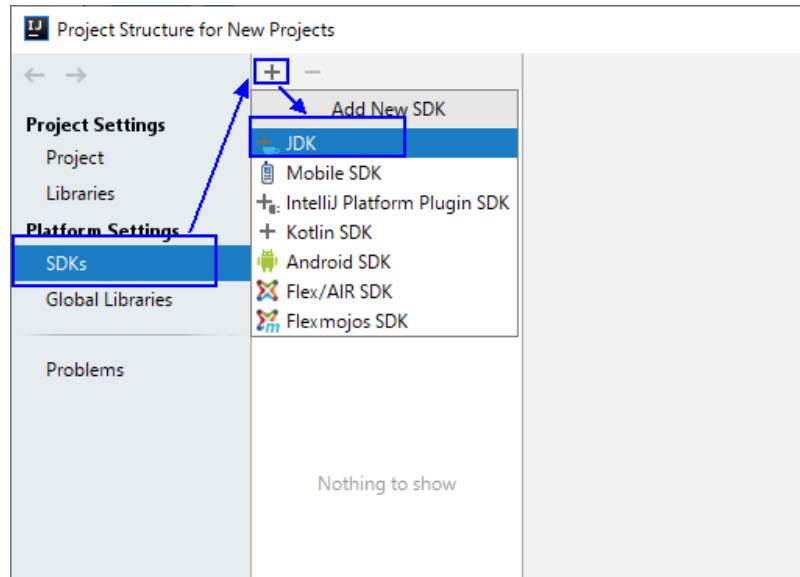
2. Select **Structure for New Projects** from the drop-down list.

Figure 3-2 Clicking Configure



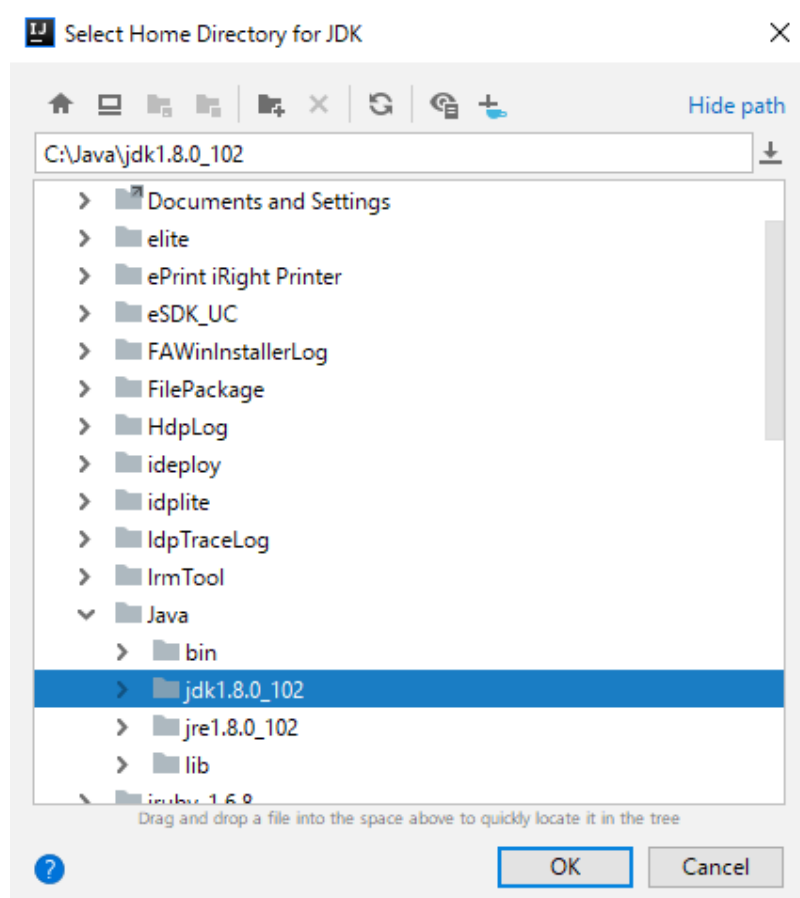
3. On the displayed **Project Structure for New Projects** page, select **SDKs**, click the plus sign (+), and click **JDK**.

Figure 3-3 Project Structure for New Projects



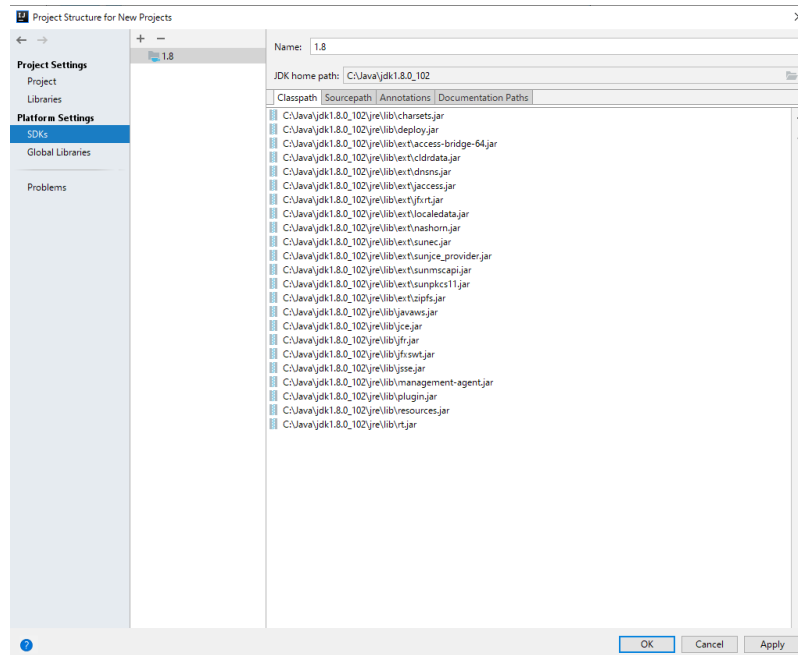
4. On the **Select Home Directory for JDK** page that is displayed, select the JDK directory and click **OK**.

Figure 3-4 Select Home Directory for JDK



5. After selecting the JDK, click **OK** to complete the configuration.

Figure 3-5 Completing the configuration



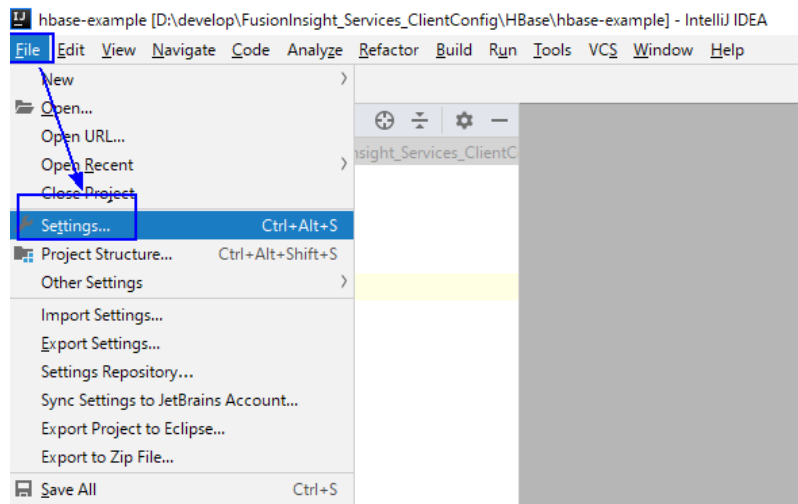
NOTE

The operation procedure may vary according to the IDEA version.

Step 2 Set the Maven version used by the project.

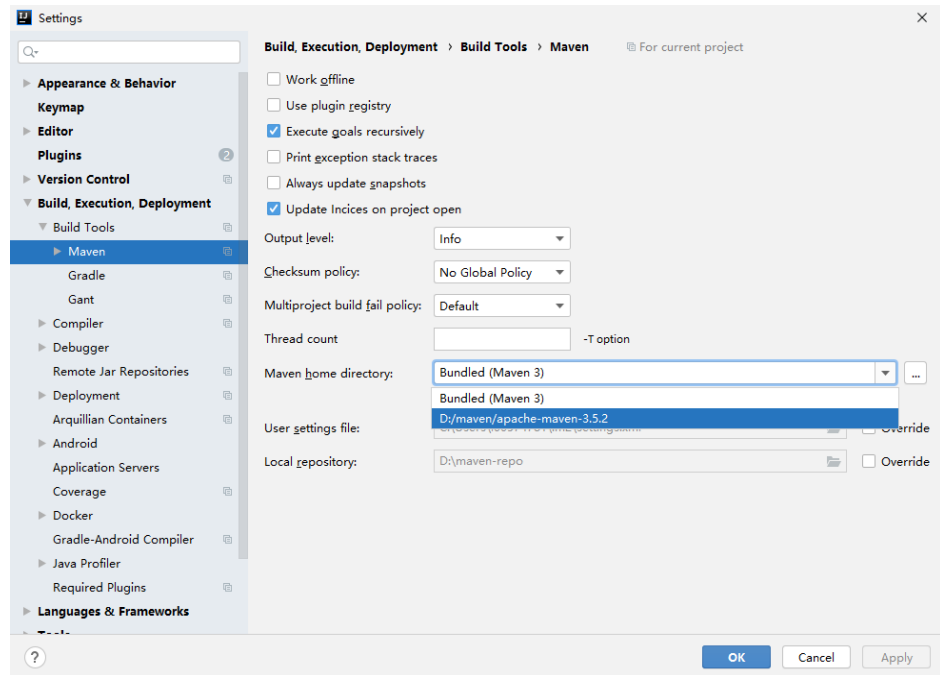
1. Choose **File > Settings...** from the main menu of IntelliJ IDEA.

Figure 3-6 Settings



2. Choose **Build, Execution, Deployment > Maven** and set **Maven home directory** to the Maven version installed on the local PC.
Set **User settings file** and **Local repository** as you need, and click **Apply > OK**.

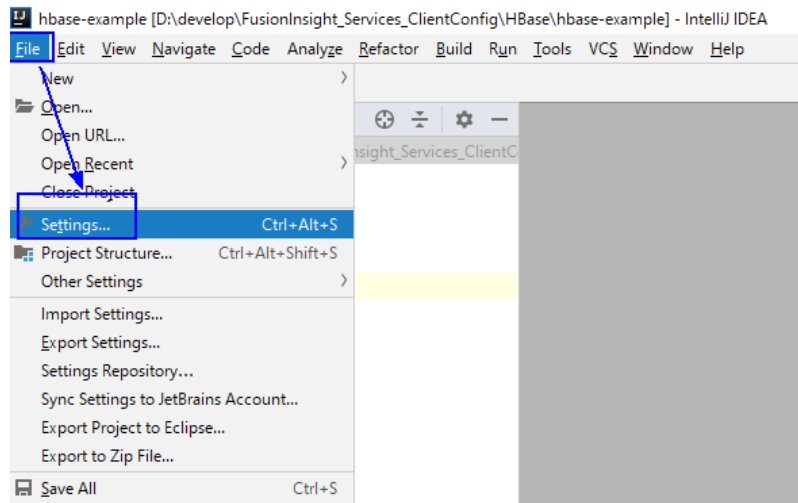
Figure 3-7 Selecting the local Maven installation directory



Step 3 Set the IntelliJ IDEA text file coding format to prevent garbled characters.

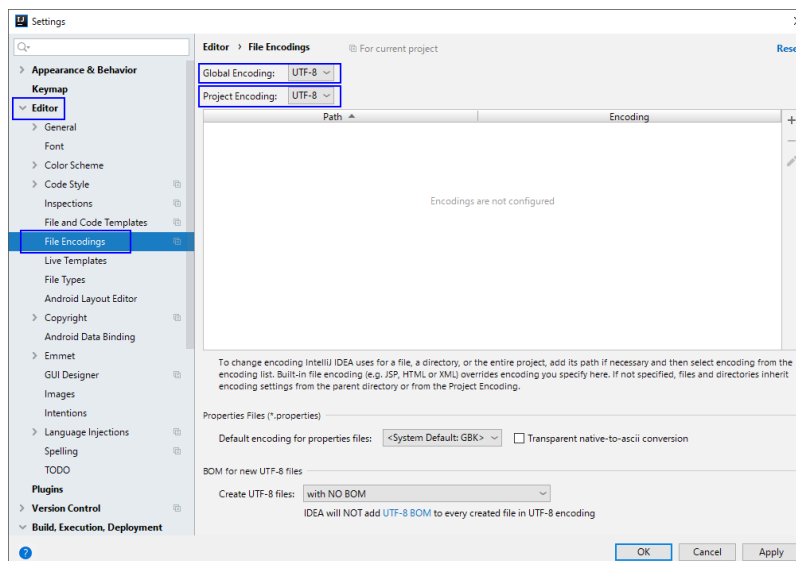
1. Choose **File > Settings...** from the main menu of IntelliJ IDEA.

Figure 3-8 Settings



2. In the navigation tree of the **Settings** page, choose **Editor > File Encodings**, and select **UTF-8** for both **Global Encoding** and **Project Encoding**.

Figure 3-9 File Encodings



3. Click **Apply** and **OK** to complete the configuration.

Step 4 Add the LakeFormation Java SDK dependencies collected in [Collecting Dependency Information](#) and the maven-assembly-plugin dependencies to the end of the **pom.xml** file of Maven.

----End

3.2 Reference Example

After the development and running environments are prepared, you can develop samples as required.

The following is an example of using LakeFormation Java SDK to develop programs:

(The following code describes how to initialize the SDK, create a LakeFormationClient instance, create a request, add parameters, and query the catalog list.)

```
package com.huawei.cloud.dalf.lakecat.examples;

import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ClientRequestException;
import com.huaweicloud.sdk.core.exception.ServerResponseException;
import com.huaweicloud.sdk.core.http.HttpConfig;
import com.huaweicloud.sdk.lakeformation.v1.LakeFormationClient;
import com.huaweicloud.sdk.lakeformation.v1.model.ListCatalogsRequest;
import com.huaweicloud.sdk.lakeformation.v1.model.ListCatalogsResponse;

import java.util.ArrayList;
import java.util.List;

public class LakeFormationExample {
    public static void main(String[] args) {
        // The getAk() and getSk() methods need to be implemented by yourself. You can obtain the AK/SK
        // from the configuration item or other locations.
        // Do not hard-code AK and SK in codes.
        String ak = getAk();
        String sk = getSk();
    }
}
```

```
// projectId: project ID
String projectId = "{*****your project id*****}";

// 1. Initialize the SDK.
HttpConfig config = HttpConfig.getDefaultHttpConfig();
config.withIgnoreSSLVerification(true);
List<String> endpoints = new ArrayList<>();
endpoints.add("lakeformation.lakecat.com");
BasicCredentials basicCredentials = new
BasicCredentials().withAk(ak).withSk(sk).withProjectId(projectId);

// 2. Create a LakeFormationClient instance.
LakeFormationClient client = LakeFormationClient.newBuilder()
.withHttpConfig(config)
.withCredential(basicCredentials)
.withEndpoints(endpoints)
.build();

// 3. Create a request and add parameters.
ListCatalogsRequest listCatalogsRequest =
new ListCatalogsRequest().withInstanceId("{*****your instance id*****}");

// 4. Query the catalog list.
try {
ListCatalogsResponse response = client.listCatalogs(listCatalogsRequest);
System.out.println(response.getHttpStatusCode());
System.out.println(response);
} catch (ClientRequestException | ServerResponseException e) {
System.out.println(e.getHttpStatusCode());
System.out.println(e.getMessage());
}
}
```

3.3 Commissioning Applications

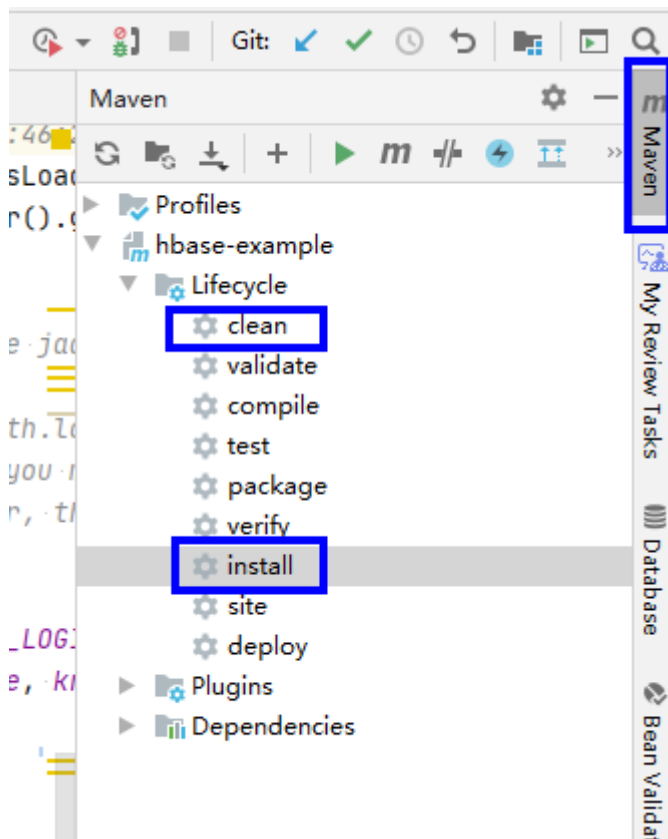
After configuring the sample code, export the JAR package and upload it to the node where the LakeFormation client is deployed.

Step 1 Export a JAR file.

Choose **Maven**, locate the target project name, and double-click **clean** under **Lifecycle** to run the **clean** command of Maven.

Choose **Maven**, locate the target project name, and double-click **install** under **Lifecycle** to run the **install** command of Maven.

Figure 3-10 Maven clean and install



Step 2 Run the **maven** command to package the project and run the following command to upload the JAR package to the node where the LakeFormation client is located:

```
java -cp lakeformation-lakecat-opensource-1.0.0-jar-with-dependencies.jar com.huawei.cloud.dalf.lakecat.examples.LakeFormationExample
```

The command output is as follows.

```
root@performance-test:~# java -cp lakeformation-lakecat-opensource-1.0.0-jar-with-dependencies.jar com.huawei.cloud.dalf.lakecat.examples.LakeFormationExample
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/manual.html#staticLoggerBinder for further details.
100
class ListCatalogsResponse {
  body: [class Catalog {
    catalogName: hive
    description: Default catalog, for Hive sa
    location: obs://lakeformation-
    databaseLocationList: null
  }, class Catalog {
    catalogName: test
    description:
    location: obs://
    databaseLocationList: null
  }, class Catalog {
    catalogName: x
    description:
    location: obs://
    databaseLocationList: null
  }]
}
```

----End