# Automating API Management

Table of Contents

[Automating API Management 1](#_Toc509915781)

[Overview 3](#_Toc509915782)

[Step 1, Login 4](#_Toc509915783)

[Documentation 4](#_Toc509915784)

[Example 4](#_Toc509915785)

[Code Snippet 4](#_Toc509915786)

[Step 2, Create an HTTP Asset in Exchange 5](#_Toc509915787)

[Documentation 5](#_Toc509915788)

[Code Snippet 5](#_Toc509915789)

[Step 3, Create an API Instance 7](#_Toc509915790)

[Documentation 7](#_Toc509915791)

[Example 7](#_Toc509915792)

[Code Snippet 7](#_Toc509915793)

[Step 4, Add Policies to API Instance 9](#_Toc509915794)

[Documentation 9](#_Toc509915795)

[Example 9](#_Toc509915796)

[Code Snippet 9](#_Toc509915797)

[Step 5, Add Alerts to API Instance 10](#_Toc509915798)

[Documentation 10](#_Toc509915799)

[Step 6, Create Client Application in Exchange 11](#_Toc509915800)

[Documentation 11](#_Toc509915801)

[Code Snippet 11](#_Toc509915802)

[Step 7, Get Client Application Access to API 12](#_Toc509915803)

[Documentation 12](#_Toc509915804)

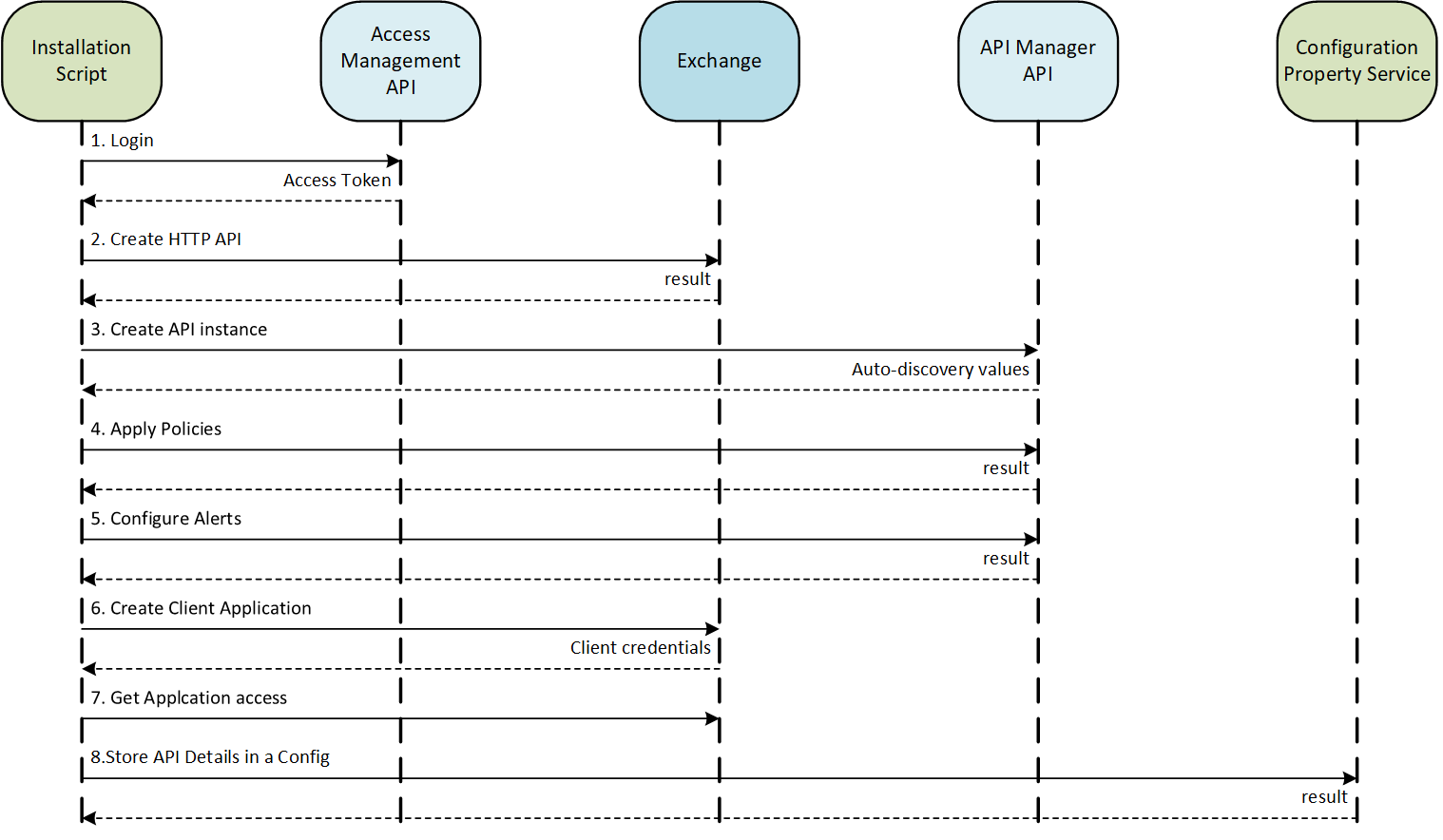
[Code Snippet 12](#_Toc509915805)

[Step 8, Store the API Details in a Config 13](#_Toc509915806)

[Example pom.xml 14](#_Toc509915807)

[Example Imports 17](#_Toc509915808)

## Overview



The purpose of this document is to provide some high-level details for how to automate the installation of a new API and its setup in API Manager with policies and client applications. At the end of the automated installation, the client application can use the new API using the credentials that were issued to it.

This automated installation will be used as part of a larger installation process that includes deployment of applications and other API implementations for a variety of platforms...including Mule applications using the Anypoint Runtime Manager (ARM).

This document leaves out many details about retrieving the data elements used to support the functions shown in the diagram above. To supplement this document, a complete example is provided as a java application (ApiDeployTool).

## Step 1, Login

Utilize the Access Management API to obtain a token that is used in the remaining API calls.

Use the /login resource to get the access token. Organization and group ids can also be obtained through this API.

### Documentation

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/access-management-api/1.0.0/console/resources/login/>

### Example

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/access-management-api/api/v1/pages/Basic%20Examples/>

### Code Snippet

@SuppressWarnings("unchecked")

private static String getAPToken(Client restClient, String user, String password) throws JsonProcessingException {

String token = null;

LinkedHashMap<String, Object> loginValues = new LinkedHashMap<String, Object>();

loginValues.put("username", user);

loginValues.put("password", password);

String payload = new ObjectMapper().writerWithDefaultPrettyPrinter().writeValueAsString(loginValues);

WebTarget target = restClient.target(HTTPS\_ANYPOINT\_MULESOFT\_COM).path("accounts/login");

Response response = target.request().accept(MediaType.APPLICATION\_JSON)

.post(Entity.entity(payload, MediaType.APPLICATION\_JSON));

int statuscode = 500;

Map<String, Object> result = null;

if (response != null) {

statuscode = response.getStatus();

}

if (response != null && response.getStatus() == 200) {

result = response.readEntity(Map.class);

token = (String) result.get("access\_token");

} else {

System.err.println("Failed to login...check credentials");

System.exit(statuscode);

}

return token;

}

## Step 2, Create an HTTP Asset in Exchange

Get the existing API definition or create a new HTTP API asset in the target business group’s Exchange. Use the Exchange Experience API to POST a payload to Exchange.

### Documentation

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/exchange-experience-api/1.0.1/console/resources/assets/>

### Code Snippet

@SuppressWarnings("unchecked")

private static void publishAPItoExchange(Client restClient, String authorizationHdr, String apiName,

String apiVersion, String organizationName, String organizationId, String groupName, String groupId)

throws JsonProcessingException {

String assetVersion = "1.0.0";

StringBuilder assetId = new StringBuilder();

assetId.append(groupId).append("\_").append(apiName).append("\_").append(assetVersion);

StringBuilder name = new StringBuilder();

name.append(apiName).append("\_").append(groupName);

WebTarget target = restClient.target(HTTPS\_ANYPOINT\_MULESOFT\_COM).path("exchange/api/v1/assets");

FormDataMultiPart form = new FormDataMultiPart();

form.setMediaType(MediaType.MULTIPART\_FORM\_DATA\_TYPE);

form.field("organizationId", groupId);

form.field("groupId", groupId);

form.field("assetId", assetId.toString());

form.field("version", assetVersion);

form.field("name", name.toString());

form.field("apiVersion", apiVersion);

form.field("classifier", "http");

form.field("asset", "undefined");

Response response = target.request().accept(MediaType.APPLICATION\_JSON)

.header("Authorization", authorizationHdr).post(Entity.entity(form, form.getMediaType()));

int statuscode = 500;

LinkedHashMap<String, Object> result = null;

if (response != null) {

statuscode = response.getStatus();

}

if (response != null && response.getStatus() == 201) {

result = response.readEntity(LinkedHashMap.class);

} else {

System.err.println("Failed to post API to Exchange. (" + statuscode + ")");

System.err.println(response.readEntity(String.class));

}

if (result != null) {

// ObjectMapper mapperw = new ObjectMapper();

// System.err.println(

// "new Exchange asset: " +

// mapperw.writerWithDefaultPrettyPrinter().writeValueAsString(result));

} else {

System.err.println("Failed to publish Exchange asset");

System.exit(statuscode);

}

}

## Step 3, Create an API Instance

Use existing or create a new instance of the API in the target business group’s environment of API Manager. Use the ../apis resource of the API Manager to do this.

### Documentation

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/api-manager-api/1.0.2/console/resources/organizations%2F%7BorganizationId%7D%2Fenvironments%2F%7BenvironmentId%7D%2Fapis/>

### Example

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/api-manager-api/api/v1/pages/Getting%20Started/>

### Code Snippet

@SuppressWarnings("unchecked")

private static void registerAPIInstance(Client restClient, String authorizationHdr, String businessGroupId,

String environmentId, String assetId, String assetVersion) throws JsonProcessingException {

HashMap<String, Object> body = new HashMap<String, Object>();

LinkedHashMap<String, Object> specValues = new LinkedHashMap<String, Object>();

specValues.put("groupId", businessGroupId);

specValues.put("assetId", assetId);

specValues.put("version", assetVersion);

body.put("spec", specValues);

body.put("instanceLabel", "auto-api-registation-" + assetId);

LinkedHashMap<String, Object> endpointValues = new LinkedHashMap<String, Object>();

endpointValues.put("uri", "https://some.implementation.com");

endpointValues.put("proxyUri", null);

endpointValues.put("isCloudHub", false);

body.put("endpoint", endpointValues);

String payload = new ObjectMapper().writerWithDefaultPrettyPrinter().writeValueAsString(body);

WebTarget target = restClient.target(HTTPS\_ANYPOINT\_MULESOFT\_COM).path("apimanager/api/v1/organizations")

.path(businessGroupId).path("environments").path(environmentId).path("apis");

Response response = target.request().accept(MediaType.APPLICATION\_JSON)

.header("Authorization", authorizationHdr).post(Entity.entity(payload, MediaType.APPLICATION\_JSON));

int statuscode = 500;

LinkedHashMap<String, Object> result = null;

if (response != null) {

statuscode = response.getStatus();

}

if (response != null && response.getStatus() == 201) {

result = response.readEntity(LinkedHashMap.class);

} else {

System.err.println("Failed to register API to API Manager. (" + statuscode + ")");

System.err.println(response.readEntity(String.class));

}

if (result != null) {

// ObjectMapper mapperw = new ObjectMapper();

// System.err.println(

// "new API instance: " + mapperw.writerWithDefaultPrettyPrinter().writeValueAsString(result));

} else {

System.err.println("Failed to create API instance");

System.exit(statuscode);

}

}

## Step 4, Add Policies to API Instance

Similarly, use API Manager ../apis/../policies resource to add policies to the API instance.

### Documentation

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/api-manager-api/1.0.2/console/resources/organizations%2F%7BorganizationId%7D%2Fenvironments%2F%7BenvironmentId%7D%2Fapis%2F%7BenvironmentApiId%7D%2Fpolicies/>

### Example

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/api-manager-api/api/v1/pages/Applying%20a%20policy/>

### Code Snippet

private static void addApiPolicy(Client restClient, String authorizationHdr, String businessGroupId,

String environmentId, String apiInstanceId, LinkedHashMap<String, Object> apiPolicy)

throws JsonProcessingException {

String policyStr = null;

try {

ObjectMapper mapperw = new ObjectMapper();

policyStr = mapperw.writeValueAsString(apiPolicy);

// System.err.println("Setting policy " + policyStr);

WebTarget target = restClient.target(HTTPS\_ANYPOINT\_MULESOFT\_COM).path("apimanager/api/v1/organizations")

.path(businessGroupId).path("environments").path(environmentId).path("apis").path(apiInstanceId)

.path("policies");

Response response = target.request().accept(MediaType.APPLICATION\_JSON)

.header("Authorization", authorizationHdr)

.post(Entity.entity(policyStr, MediaType.APPLICATION\_JSON));

int statuscode = 500;

if (response != null) {

statuscode = response.getStatus();

}

if (response != null && (response.getStatus() == 201 || response.getStatus() == 409)) {

// System.err.println(response.readEntity(String.class));

} else {

System.err.println("Failed to apply policy " + policyStr + ". (" + statuscode + ")");

System.err.println(response.readEntity(String.class));

}

} catch (Exception e) {

System.err.println("Cannot set policy:\n " + policyStr);

e.printStackTrace(System.err);

}

}

## Step 5, Add Alerts to API Instance

Use API Manager ../apis/../alerts resource to add alerts to the API instance.

### Documentation

<https://anypoint.mulesoft.com/exchange/portals/anypoint-platform-eng/f1e97bc6-315a-4490-82a7-23abe036327a.anypoint-platform/api-manager-api/1.0.2/console/resources/organizations%2F%7BorganizationId%7D%2Fenvironments%2F%7BenvironmentId%7D%2Fapis%2F%7BenvironmentApiId%7D%2Falerts/>

## Step 6, Create Client Application in Exchange

Create one or more client applications in Exchange that will be accessing the API.

### Documentation

There is no documentation for this function. Use the Exchange Experience API to perform this function.

### Code Snippet

private static void createApplication(Client restClient, String authorizationHdr,

String organizationId, String applicationName, String description) throws JsonProcessingException {

String desc = (description == null)

? "Auto generated client credentials for this API instance to use calling other dependencies."

: description;

LinkedHashMap<String, Object> applicationValues = new LinkedHashMap<String, Object>();

applicationValues.put("name", applicationName);

applicationValues.put("description", desc);

applicationValues.put("redirectUri", new ArrayList<String>());

applicationValues.put("grantTypes", new ArrayList<String>());

applicationValues.put("apiEndpoints", false);

String payload = new ObjectMapper().writerWithDefaultPrettyPrinter().writeValueAsString(applicationValues);

WebTarget target = restClient.target(HTTPS\_ANYPOINT\_MULESOFT\_COM).path("exchange/api/v1/organizations")

.path(organizationId).path("applications");

Response response = target.request().header("Authorization", authorizationHdr)

.accept(MediaType.APPLICATION\_JSON).post(Entity.entity(payload, MediaType.APPLICATION\_JSON));

int statuscode = 500;

if (response != null) {

statuscode = response.getStatus();

}

if (response != null && (response.getStatus() == 201 || response.getStatus() == 409)) {

// System.err.println(response.readEntity(String.class));

} else {

System.err.println("Failed to create application information (" + statuscode + ")");

System.err.println(response.readEntity(String.class));

System.exit(statuscode);

}

}

## Step 7, Get Client Application Access to API

Grant access to the client application to use the API.

### Documentation

There is no documentation for this function. Use the Exchange Experience API to perform this function.

### Code Snippet

private static void createApplicationContract(Client restClient, String authorizationHdr,

String organizationId, int applicationId, String businessGroupId,

String environmentId, String exchangeAssetId, String exchangeAssetVersion, String autoDiscoveryApiId, String apiVersion) throws JsonProcessingException {

LinkedHashMap<String, Object> contractValues = new LinkedHashMap<String, Object>();

contractValues.put("apiId", autoDiscoveryApiId);

contractValues.put("environmentId", environmentId);

contractValues.put("acceptedTerms", true);

contractValues.put("organizationId", businessGroupId);

contractValues.put("groupId", businessGroupId);

contractValues.put("assetId", exchangeAssetId);

contractValues.put("version", exchangeAssetVersion);

contractValues.put("productAPIVersion", apiVersion);

String payload = new ObjectMapper().writerWithDefaultPrettyPrinter().writeValueAsString(contractValues);

WebTarget target = restClient.target(HTTPS\_ANYPOINT\_MULESOFT\_COM).path("exchange/api/v1/organizations")

.path(organizationId).path("applications").path(Integer.toString(applicationId)).path("contracts");

Response response = target.request().header("Authorization", authorizationHdr)

.accept(MediaType.APPLICATION\_JSON).post(Entity.entity(payload, MediaType.APPLICATION\_JSON));

int statuscode = 500;

if (response != null) {

statuscode = response.getStatus();

}

if (response != null && (response.getStatus() == 201 || response.getStatus() == 409)) {

// System.err.println(response.readEntity(String.class));

} else {

System.err.println("Failed to create application contract (" + statuscode + ")");

System.err.println(response.readEntity(String.class));

}

}

## Step 8, Store the API Details in a Config

A configuration property service will be used to store the API instance particulars for use in configuring other applications that utilize this API instance.

# Example pom.xml

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.mulesoft.java</groupId>

<artifactId>mule-api-deploy</artifactId>

<version>1.0.0</version>

<name>Mule API Deployment Tool</name>

<description>Deploy a Mule API to Anypoint Platform</description>

<properties>

<jdk.version>1.8</jdk.version>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<jackson-2-version>2.9.3</jackson-2-version>

<mule.version>3.9.0</mule.version>

<jersey-client-version>1.19.3</jersey-client-version>

</properties>

<build>

<finalName>ApiDeployTool</finalName>

<plugins>

<!-- download source code in Eclipse, best practice -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-eclipse-plugin</artifactId>

<version>2.9</version>

<configuration>

<downloadSources>true</downloadSources>

<downloadJavadocs>false</downloadJavadocs>

</configuration>

</plugin>

<!-- Set a compiler level -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>2.3.2</version>

<configuration>

<source>${jdk.version}</source>

<target>${jdk.version}</target>

</configuration>

</plugin>

<!-- Maven Shade Plugin -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-shade-plugin</artifactId>

<version>2.3</version>

<executions>

<!-- Run shade goal on package phase -->

<execution>

<phase>package</phase>

<goals>

<goal>shade</goal>

</goals>

<configuration>

<transformers>

<!-- add Main-Class to manifest file -->

<transformer

implementation="org.apache.maven.plugins.shade.resource.ManifestResourceTransformer">

<mainClass>com.mulesoft.java.ApiDeployTool</mainClass>

</transformer>

</transformers>

</configuration>

</execution>

</executions>

</plugin>

</plugins>

</build>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<!-- https://mvnrepository.com/artifact/commons-io/commons-io -->

<dependency>

<groupId>commons-io</groupId>

<artifactId>commons-io</artifactId>

<version>2.5</version>

</dependency>

<!-- the core, which includes Streaming API, shared low-level abstractions

(but NOT data-binding) -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-core</artifactId>

<version>${jackson-2-version}</version>

</dependency>

<!-- Just the annotations; use this dependency if you want to attach annotations

to classes without connecting them to the code. -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-annotations</artifactId>

<version>${jackson-2-version}</version>

</dependency>

<!-- databinding; ObjectMapper, JsonNode and related classes are here -->

<dependency>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<version>${jackson-2-version}</version>

</dependency>

<dependency>

<groupId>com.fasterxml.jackson.jaxrs</groupId>

<artifactId>jackson-jaxrs-json-provider</artifactId>

<version>${jackson-2-version}</version>

</dependency>

<!-- https://mvnrepository.com/artifact/commons-logging/commons-logging -->

<dependency>

<groupId>commons-logging</groupId>

<artifactId>commons-logging</artifactId>

<version>1.2</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.glassfish.jersey.core/jersey-client -->

<dependency>

<groupId>org.glassfish.jersey.core</groupId>

<artifactId>jersey-client</artifactId>

<version>2.26</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.glassfish.jersey.inject/jersey-hk2 -->

<dependency>

<groupId>org.glassfish.jersey.inject</groupId>

<artifactId>jersey-hk2</artifactId>

<version>2.26</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.glassfish.jersey.media/jersey-media-multipart -->

<dependency>

<groupId>org.glassfish.jersey.media</groupId>

<artifactId>jersey-media-multipart</artifactId>

<version>2.26</version>

</dependency>

<!-- Mule core is used for IOUtils -->

<!-- https://mvnrepository.com/artifact/org.mule/mule-core -->

<dependency>

<groupId>org.mule</groupId>

<artifactId>mule-core</artifactId>

<version>${mule.version}</version>

</dependency>

<dependency>

<groupId>org.mule.consulting</groupId>

<artifactId>cps-encryption</artifactId>

<version>1.1.0</version>

</dependency>

</dependencies>

</project>

# Example Imports

import java.io.File;

import java.io.InputStream;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.LinkedHashMap;

import java.util.Map;

import javax.ws.rs.client.Client;

import javax.ws.rs.client.ClientBuilder;

import javax.ws.rs.client.Entity;

import javax.ws.rs.client.WebTarget;

import javax.ws.rs.core.MediaType;

import javax.ws.rs.core.Response;

import org.apache.commons.io.FileUtils;

import org.apache.commons.io.IOUtils;

import org.glassfish.jersey.media.multipart.FormDataMultiPart;

import org.glassfish.jersey.media.multipart.MultiPartFeature;

import org.mule.consulting.cps.encryption.CpsEncryptor;

import com.fasterxml.jackson.core.JsonProcessingException;

import com.fasterxml.jackson.databind.ObjectMapper;

import com.fasterxml.jackson.databind.type.CollectionType;

import com.fasterxml.jackson.databind.type.MapType;

import com.fasterxml.jackson.databind.type.TypeFactory;

import com.fasterxml.jackson.jaxrs.json.JacksonJsonProvider;