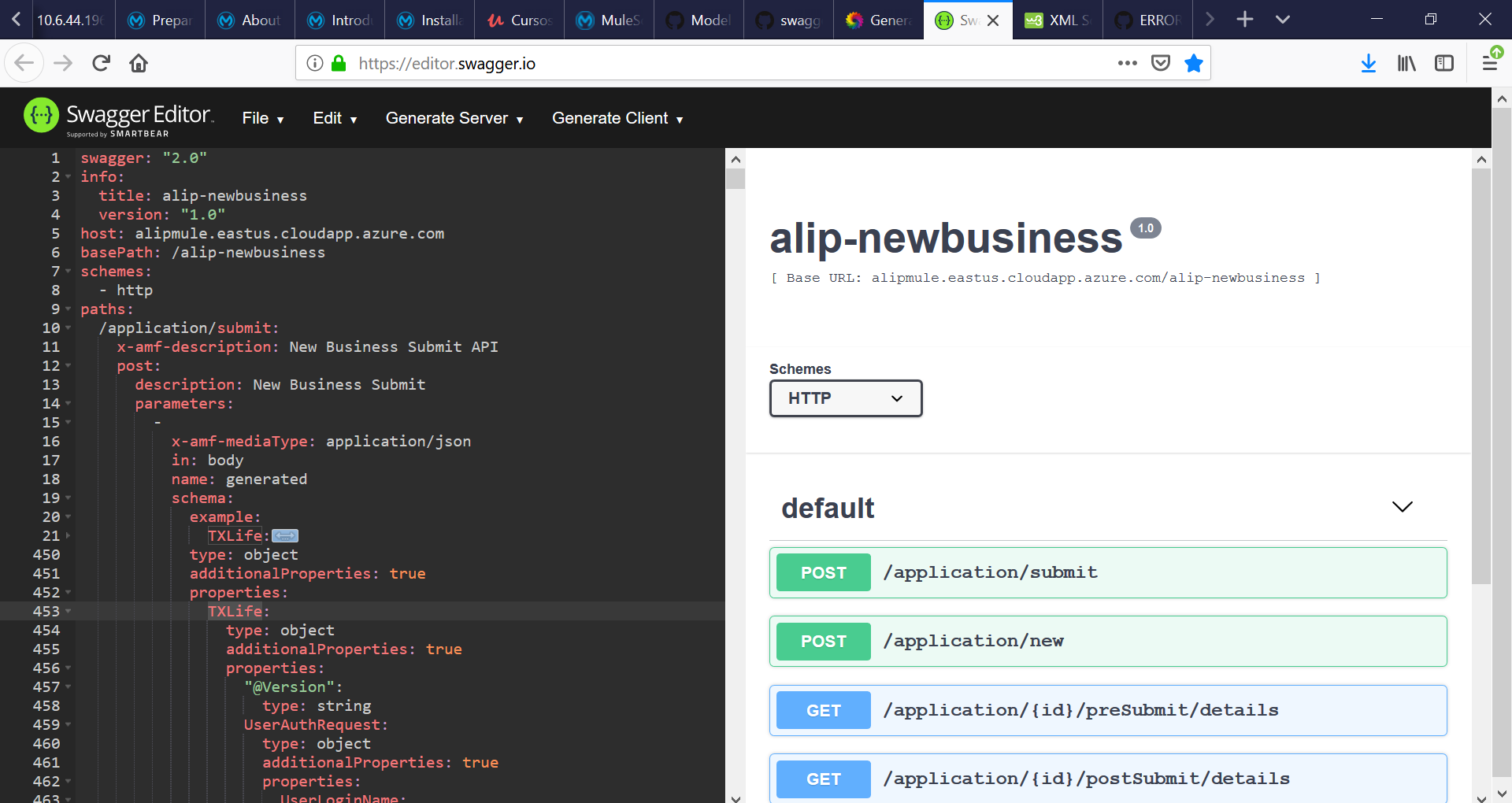
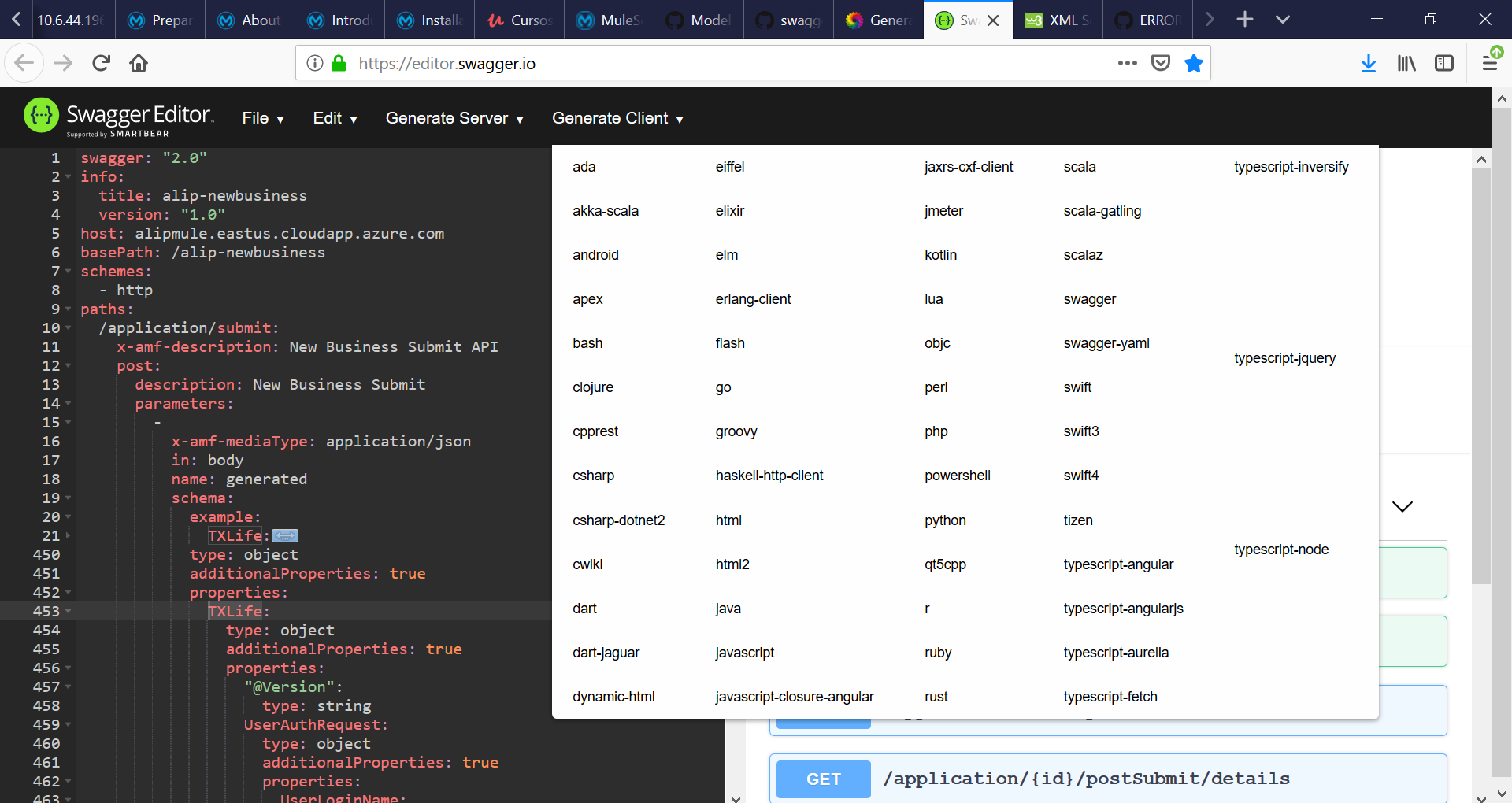
**Swagger java client generator**

– online tool: <https://editor.swagger.io/>

Benefits:

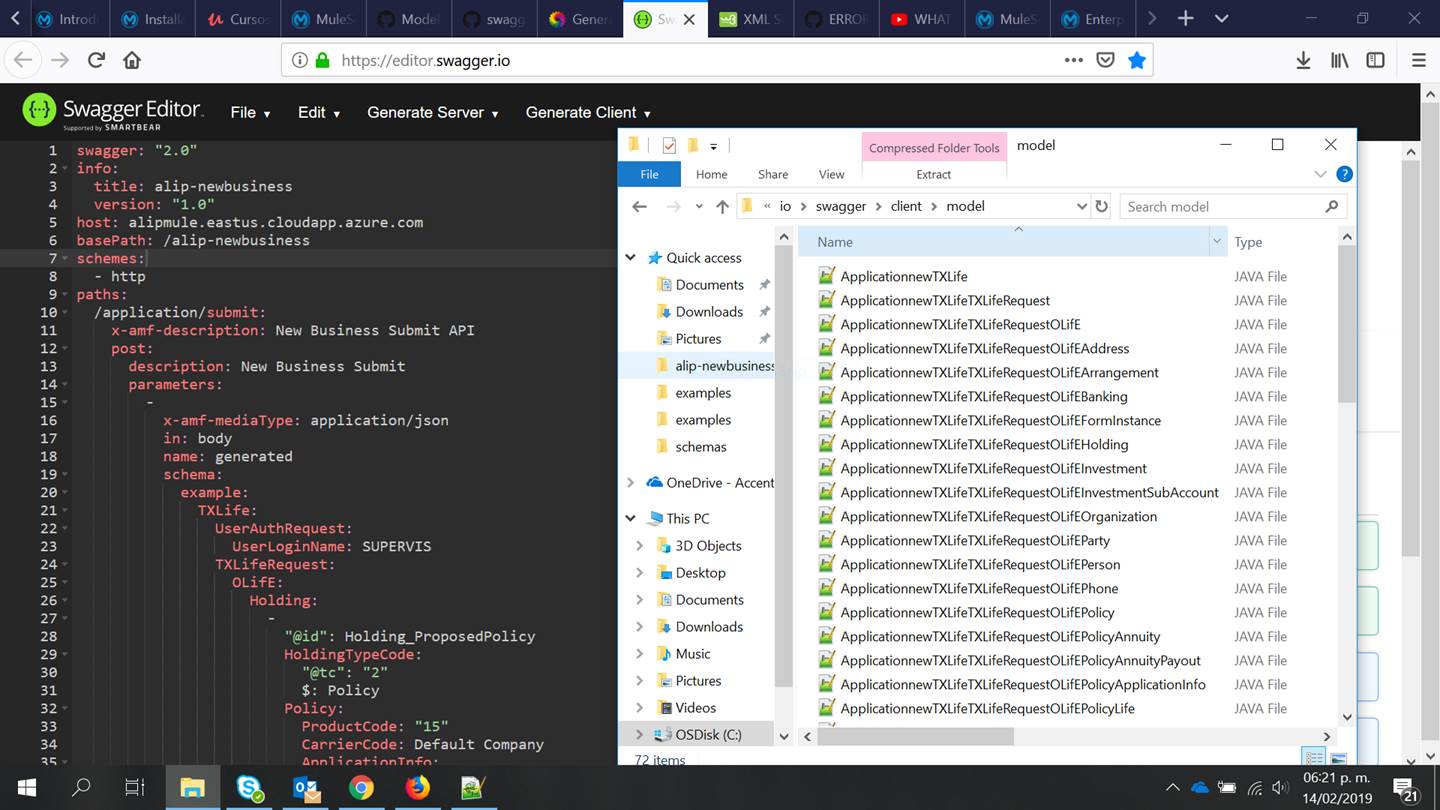
* Simplicity: nothing to download, just copy and one click to get the java client.
* Swagger code validation
* Most popular languages supported



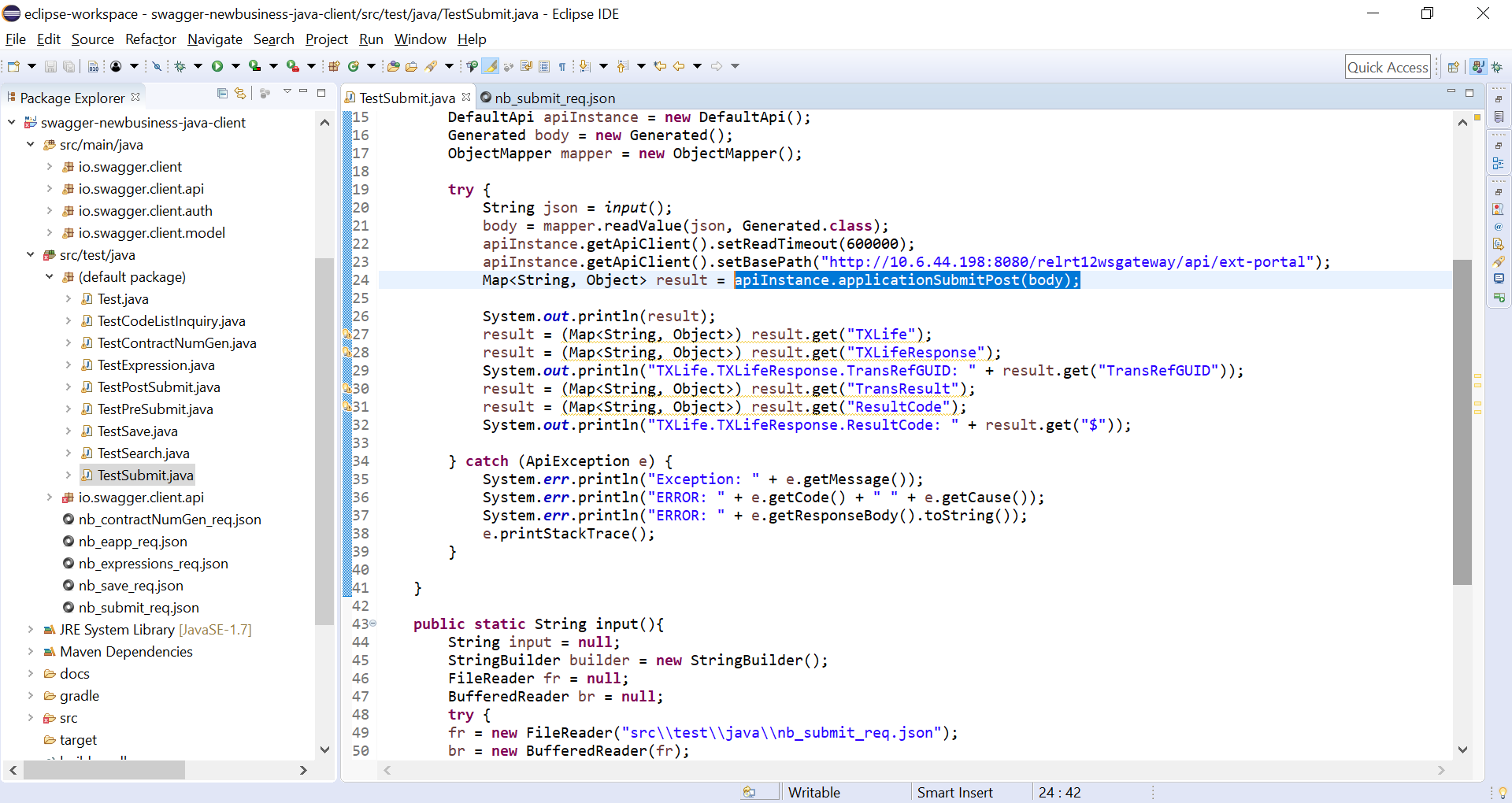


**Instructions**: Once you have a well-formed (OAS 2.0) swagger file\*\*, with the schema definitions included, copy and paste the swagger code (or drag and drop the swagger file in yaml or json format) into the **editor.swagger.io** tool to generate the java client. The client zip package will be downloaded, just verify the client and the model classes were properly generated.

*\*\* swagger file can be generated from Anypoint Platform - Design Center – API RAML definition.*



Unzip the client package and import into eclipse as a Maven project



**Test example code**

**import** java.io.BufferedReader;

**import** java.io.FileReader;

**import** java.io.IOException;

**import** java.util.Map;

**import** io.swagger.client.ApiException;

**import** io.swagger.client.api.DefaultApi;

**import** io.swagger.client.model.\*;

**import** com.fasterxml.jackson.databind.ObjectMapper;

**public** **class** TestSubmit {

**public** **static** **void** main(String[] args) **throws** IOException {

DefaultApi apiInstance = **new** DefaultApi();

Generated body = **new** Generated();

ObjectMapper mapper = **new** ObjectMapper();

**try** {

String json = *input*();

body = mapper.readValue(json, Generated.**class**);

apiInstance.getApiClient().setReadTimeout(600000);

apiInstance.getApiClient()

.setBasePath("http://10.6.44.198:8080/relrt12wsgateway/api/ext-portal");

**Map<String, Object> result = apiInstance.applicationSubmitPost(body);**

System.***out***.println(result);

result = (Map<String, Object>) result.get("TXLife");

result = (Map<String, Object>) result.get("TXLifeResponse");

System.***out***.println("TXLife.TXLifeResponse.TransRefGUID: " +

result.get("TransRefGUID"));

result = (Map<String, Object>) result.get("TransResult");

result = (Map<String, Object>) result.get("ResultCode");

System.***out***.println("TXLife.TXLifeResponse.ResultCode: " + result.get("$"));

} **catch** (ApiException e) {

System.***err***.println("Exception: " + e.getMessage());

System.***err***.println("ERROR: " + e.getCode() + " " + e.getCause());

System.***err***.println("ERROR: " + e.getResponseBody().toString());

e.printStackTrace();

}

}

**public** **static** String input(){

String input = **null**;

StringBuilder builder = **new** StringBuilder();

FileReader fr = **null**;

BufferedReader br = **null**;

**try** {

fr = **new** FileReader("src\\test\\java\\nb\_submit\_req.json");

br = **new** BufferedReader(fr);

String sCurrentLine;

**while** ((sCurrentLine = br.readLine()) != **null**) {

//System.out.println(sCurrentLine);

builder.append(sCurrentLine);

}

fr.close();

br.close();

} **catch** (IOException e) {

e.printStackTrace();

}

//System.out.println("input: " + input);

input = builder.toString();

**return** input;

}

}

**Troubleshoots**

**ERRORUNKNOWN**

Replace the wrong generated class io.swagger.client.model.ERRORUNKNOWN with **Object** class to be able to do the service call. Example:

**public** Map<String, Object> applicationSubmitPost(Generated generated) **throws** ApiException {

**Design Center (Anypoint Platform) is not including the schema definition in the OAS swagger yaml/json file generated.**

In order to get the swagger generated properly, have fixed a lot of mismatches between examples and schemas for submit and save request:

Schema says type **array**, but in example is an **object** for several fields, so changed the **example** to be an **array**.

Schema says type **number** or **integer**, but the example comes with numbers as strings, so changed schema type to **string**.

Schema says fields are **required**, but the examples doesn’t contain that required fields, so changed schema to be required as false.

Also, included following header in the top of schema files, in order to enable the swagger schema generation from design center:

{

              "$schema": "[http://json-schema.org/draft-03/schema#](http://json-schema.org/draft-03/schema)",

              "definitions": {},

              "id": "<http://example.com/example.json>",

              "properties": {

                            "TXLife": {

                                          "id": "/properties/TXLife",

                                          "properties": {

----------------------------------------------------------------------------------------------------------------------------

**Timeout exception**

Include timeout fix:

apiInstance.getApiClient().setReadTimeout(600000);

**How to populate request body example:**

**import** com.fasterxml.jackson.databind.ObjectMapper;

ObjectMapper mapper = **new** ObjectMapper();

Generated body = mapper.readValue(json, Generated.**class**);

* Include maven dependency in pom.xml

<dependency>

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

<artifactId>jackson-module-jsonSchema</artifactId>

<version>2.8.3</version>

</dependency>