

Publisher APIs are built over **PROVIDER APIs** so that the **JSON** structure and field values of **PROVIDER API** request and response can be modified as per the end user requirement at **IRIS layer**.

1. From the provider APIs **api-docs**, take the **swagger JSON**, **modify as per your requirement of JSON structure** and **save it**.

```

1 {
2   "swagger" : "2.0",
3   "info" : {
4     "description" : "AcBalance",
5     "version" : "v1.0.0",
6     "title" : "AcBal"
7   },
8   "host" : "localhost:9089",
9   "basePath" : "/api/v1.0.0/",
10  "tags" : [ ],
11  "schemes" : [ "http", "https" ],
12  "security" : [ {
13    "basicAuth" : [ ]
14  } ], {
15    "apiKey" : [ ]
16  } ],
17  "paths" : {
18    "/party/accounts/{accountId}" : {
19      "get" : {
20        "tags" : [ ],
21        "operationId" : "getAcBal",
22        "produces" : [ "application/json" ],
23        "parameters" : [ {
24          "name" : "accountId",
25          "in" : "path",
26          "description" : "Identifier of the account. Often ref",
27          "required" : true,
28          "type" : "string"

```

Here, the path is changed only.



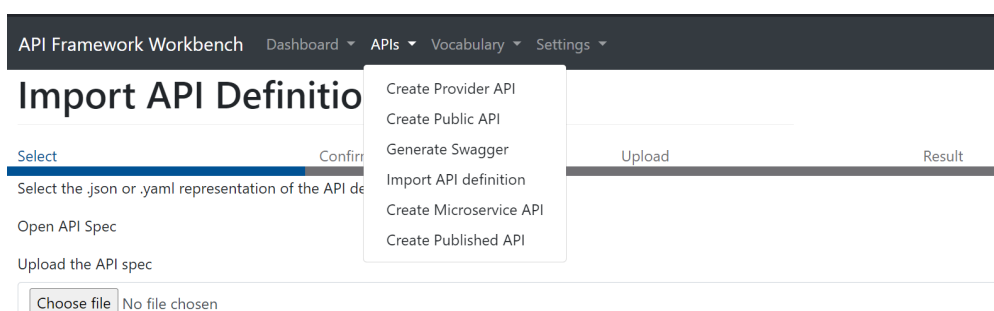
mb-api-publish-v1.0.0-swagger.json

Modified JSON File:

2. Check the correctness of that **swagger JSON** via the open-source swagger editor, the **URL** of which is given below. If any structural errors are thrown, resolve it.

<https://editor-next.swagger.io/>

3. Import this swagger definition in **workbench** using the **"Import API definition"** option and download the publisher zip file.



Import API Definition

[Previous](#)

Select	Confirm	Upload	Result
File	Type	Size	Last Modified
mb-api-publish-v1.0.0-swagger.json	application/json	10.2 KB	17/02/2024

[Upload](#)

Select	Confirm	Upload	Result
API Title	AcBal		
API Version	v1.0.0		
Service Id			

[Upload & Generate!](#)

Generating Service XML

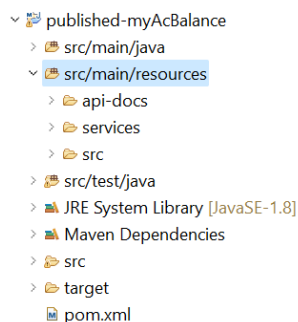
Process complete

Click the link to download the generated artefacts

[Download](#)

Download the zip file.

4. Create a maven project with **archetype-service catalog** and import the zip file downloaded in the previous step.

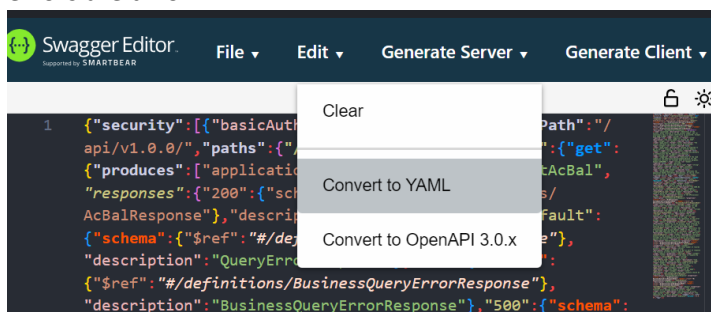


Special Note: We can skip STEP 5,6,7

5. Generate code for the **published API swagger JSON**. To generate code, do the following steps.

a. Launch <https://editor.swagger.io/> in browser.

b. Paste the content of the published API swagger JSON file. (It will be converted to YAML). Ensure no errors are thrown.



The image shows the Swagger Editor interface. On the left, the OpenAPI specification is displayed in a code editor. The specification includes a security section with basicAuth and apiKey, a basePath of /api/v1.0.0/, and a path /party/accounts/{accountId} with a GET method. The response for status 200 is defined as AcBalResponse with a schema that includes accountId, accountIBAN, currency, and availableBalance. On the right, the 'Responses' panel shows the 200 response with its description and an example value in JSON format.

```

1 security:
2   - basicAuth: []
3   - apiKey: []
4 basePath: /api/v1.0.0/
5 paths:
6   /party/accounts/{accountId}:
7     get:
8       produces:
9         - application/json
10      operationId: getAcBal
11      responses:
12        '200':
13          schema:
14            $ref: '#/definitions/AcBalResponse'
15          description: AcBalResponse
16        '400':
17          schema:
18            $ref: '#/definitions/BusinessQueryErrorResponse'
19          description: BusinessQueryErrorResponse
20        '500':
21          schema:
22            $ref: '#/definitions/SystemQueryErrorResponse'
23          description: SystemQueryErrorResponse
24      default:
25        schema:

```

Responses

Response content type: application/json

Code	Description
200	AcBalResponse

Example Value | Model

```

{
  "body": {
    {
      "accountId": "string",
      "accountIBAN": "string",
      "currency": "string",
      "availableBalance": 25
    }
  },
  "header": {
    "total_size": 0,
    "page_token": "string",
    "audit": {
      "T24_time": 0,
      "versionNumber": "string",
      "requestParse_time": 0,
      "responseParse_time": 0
    },
    "status": "string",
    "page_size": 0,
    "page_start": 0
  }
}

```

c. Click the option “Generate Client” and select “java” as shown in the below image.

The image shows the Swagger Editor interface with the 'Generate Client' dropdown menu open. The menu lists various programming languages and frameworks, with 'java' highlighted. The background shows the same OpenAPI specification as in the previous image.

```

1 security:
2   - basicAuth: []
3   - apiKey: []
4 basePath: /api/v1.0.0/
5 paths:
6   /party/accounts/{accountId}:
7     get:
8       produces:
9         - application/json
10      operationId: getAcBal
11      responses:
12        '200':
13          schema:
14            $ref: '#/definitions/AcBalResponse'
15          description: AcBalResponse
16        '400':
17          schema:
18            $ref: '#/definitions/BusinessQueryErrorResponse'
19          description: BusinessQueryErrorResponse
20        '500':
21          schema:
22            $ref: '#/definitions/SystemQueryErrorResponse'
23          description: SystemQueryErrorResponse
24      default:
25        schema:

```

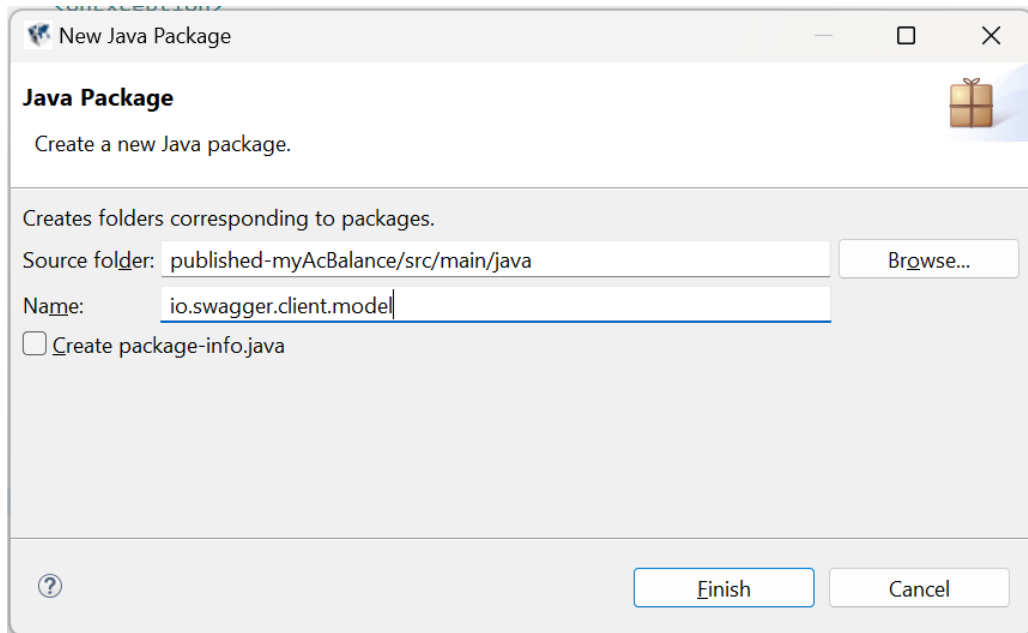
Generate Client

- ada
- akka-scala
- android
- apex
- bash
- clojure
- cpprest
- csharp
- csharp-dotnet2
- cwiki
- dart
- dart-jaguar
- dynamic-html
- eiffel
- elixir
- elm
- erlang-client
- flash
- go
- groovy
- haskell-http-client
- html
- html2
- java
- javascript
- javascript-closure-angular
- jaxrs-cxf-client
- jmeter
- kotlin
- lua
- objc
- perl
- php
- powershell
- python
- qt5cpp
- r
- ruby
- rust
- scala
- scalaz
- swagger
- swagger-yaml
- swift
- swift3
- swift4
- swift5
- tizen

d. A file named **java-client-generated.zip** will be downloaded. **Extract** the contents of this file to a folder.

e. Under **<<PublishedApi_project>>\src\main\ java** directory, create a **java package** named **io.swagger.client.model**

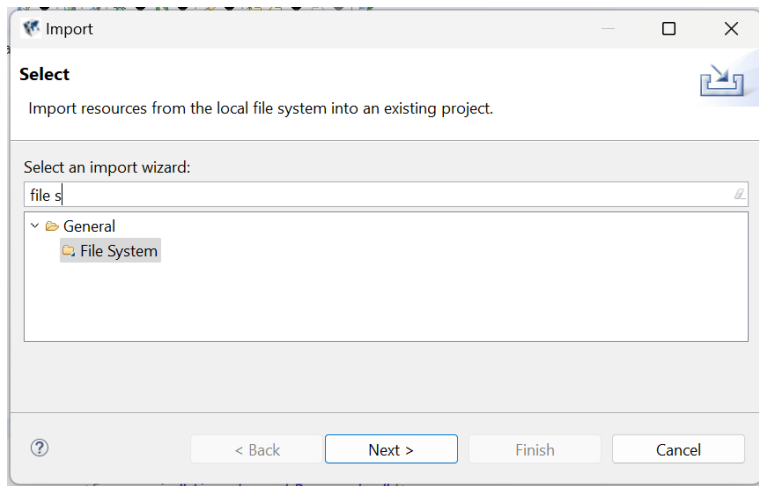
- published-myAcBalance
 - src/main/java
 - com.temenos.nazihar.published_myAcBalance
 - CustomOrchestrationRequestMapper.java



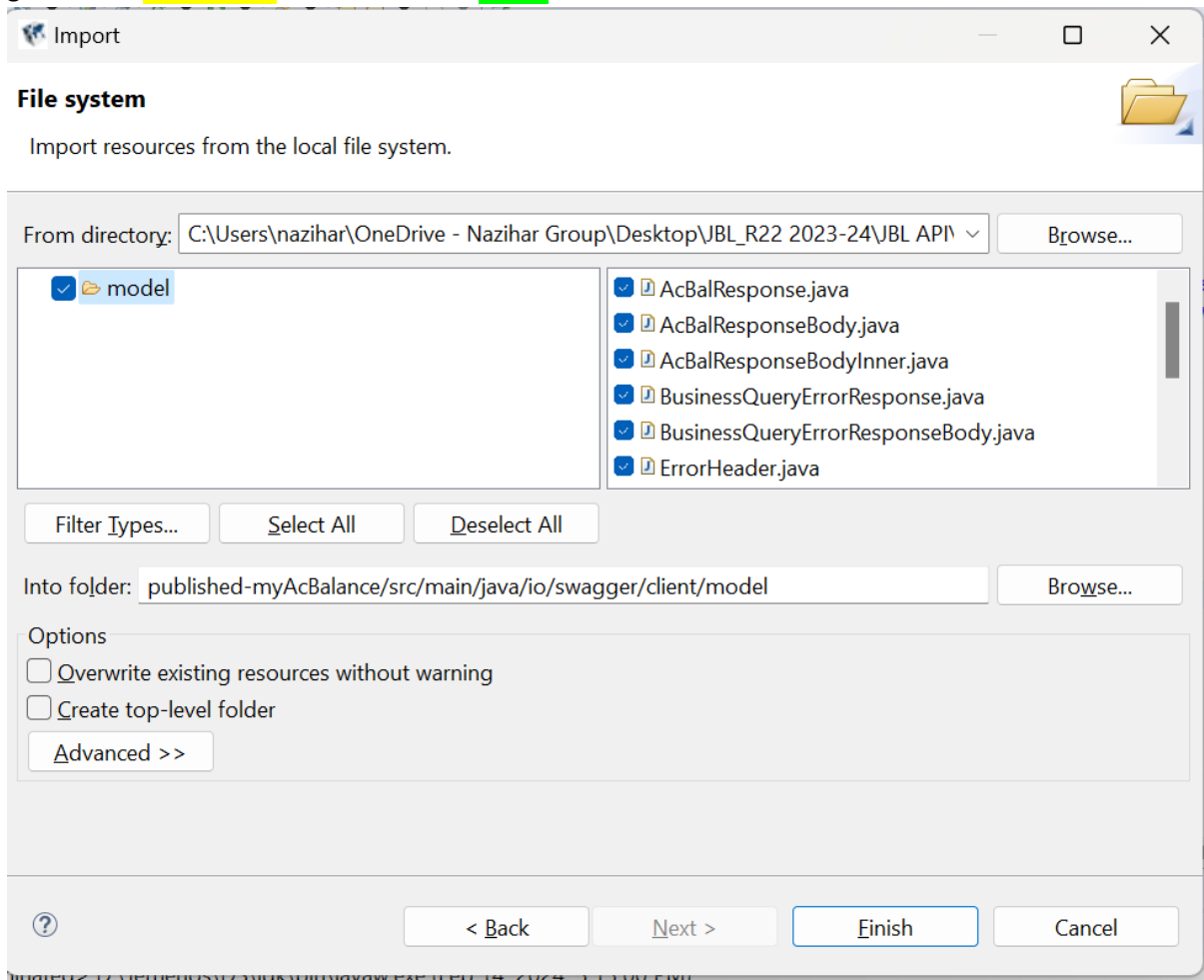
f. Right click `src/main/java/io.swagger.client.model` directory and select **Import > File System** -> navigate to the extracted zip file path `io\swagger\client\model` and select this folder.

C:\Users\nazihar\OneDrive - Nazihar Group\Desktop\JBL_R22 2023-24\JBL API\All API Resources\published-myAcBal\java-client-generated\java-client\src\main\java\io\swagger\client\model

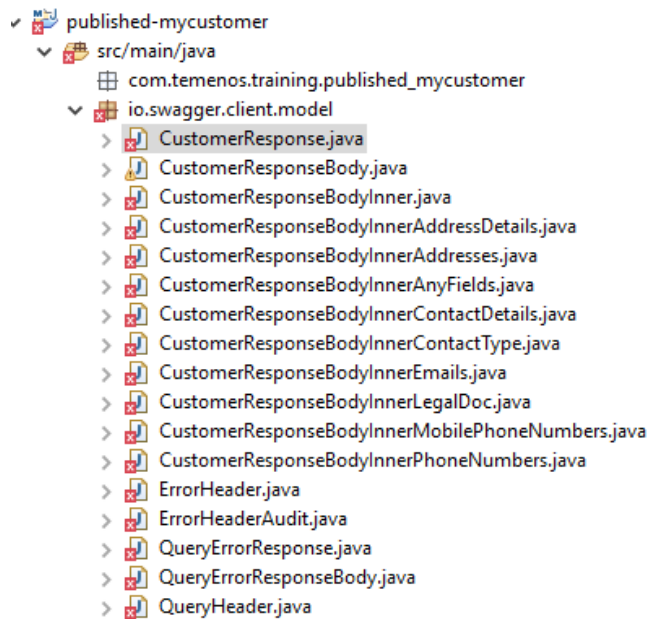
C:\Users\nazihar\OneDrive - Nazihar Group\Desktop\JBL_R22 2023-24\JBL API\All API Resources\published-myAcBal\j					
Name	Status	Date modified	Type	Size	
AcBalResponse.java	🔄	14/02/2024 9:38 AM	Java Source File	3 KB	
AcBalResponseBody.java	🔄	14/02/2024 9:38 AM	Java Source File	2 KB	
AcBalResponseBodyInner.java	🔄	14/02/2024 9:38 AM	Java Source File	5 KB	
BusinessQueryErrorResponse.java	🔄	14/02/2024 9:38 AM	Java Source File	3 KB	
BusinessQueryErrorResponseBody.java	🔄	14/02/2024 9:38 AM	Java Source File	4 KB	
ErrorHeader.java	🔄	14/02/2024 9:38 AM	Java Source File	3 KB	
ErrorHeaderAudit.java	🔄	14/02/2024 9:38 AM	Java Source File	5 KB	
QueryErrorResponse.java	🔄	14/02/2024 9:38 AM	Java Source File	3 KB	
QueryErrorResponseBody.java	🔄	14/02/2024 9:38 AM	Java Source File	4 KB	
QueryHeader.java	🔄	14/02/2024 9:38 AM	Java Source File	6 KB	
SystemQueryErrorResponse.java	🔄	14/02/2024 9:38 AM	Java Source File	3 KB	
SystemQueryErrorResponseBody.java	🔄	14/02/2024 9:38 AM	Java Source File	4 KB	



g. Click the **"Select All"** button and click **Finish**.

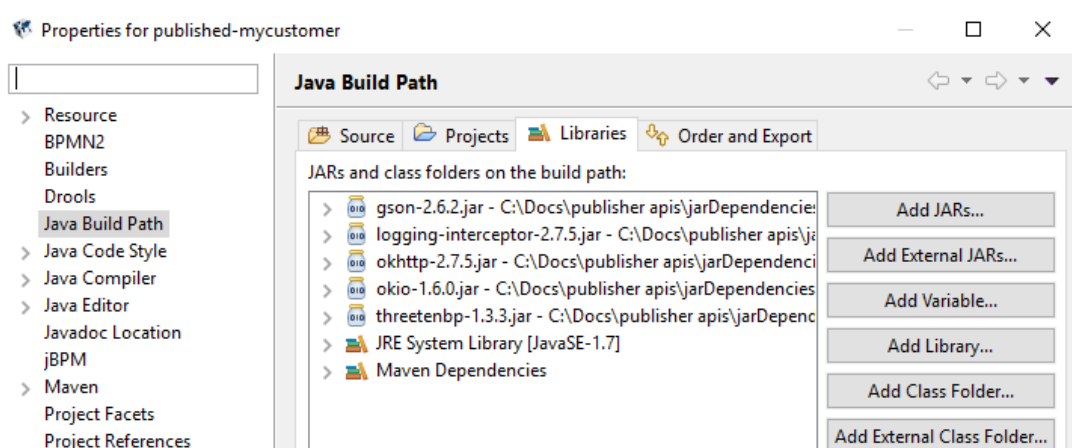


7. The imported java classes will throw error citing missing dependency jars. (DEPENDS on the SET-UP) – **May not Occur in NEW MB.**



8. **To** resolve these errors, download the below mentioned open-source jars (You can download it from any open-source jar download sites) and add it to the Build Path of the publisher API project (by right clicking the project > Configure Build Path > Libraries tab > Add External JARS > browse for the jars downloaded and select). **May not Occur in NEW MB.**

gson-2.6.2.jar	9/6/2019 12:46 PM	Executable Jar File	225 KB
logging-interceptor-2.7.5.jar	9/6/2019 12:46 PM	Executable Jar File	8 KB
okhttp-2.7.5.jar	9/6/2019 12:46 PM	Executable Jar File	324 KB
okio-1.6.0.jar	9/6/2019 12:46 PM	Executable Jar File	65 KB
threetenbp-1.3.3.jar	11/18/2019 2:13 PM	Executable Jar File	500 KB



9. Edit the **pom.xml** of the publisher API project by **adding the following dependencies.**

```
<dependency>
```

```
<groupId>com.squareup.okhttp</groupId>
```

```

<artifactId>okhttp</artifactId>

<version>2.7.5</version>

</dependency>

<dependency>

    <groupId>com.squareup.okhttp</groupId>

    <artifactId>logging-interceptor</artifactId>

    <version>2.7.5</version>

</dependency>

<dependency>

    <groupId>com.google.code.gson</groupId>

    <artifactId>gson</artifactId>

    <version>2.6.2</version>

</dependency>

<dependency>

    <groupId>org.threeten</groupId>

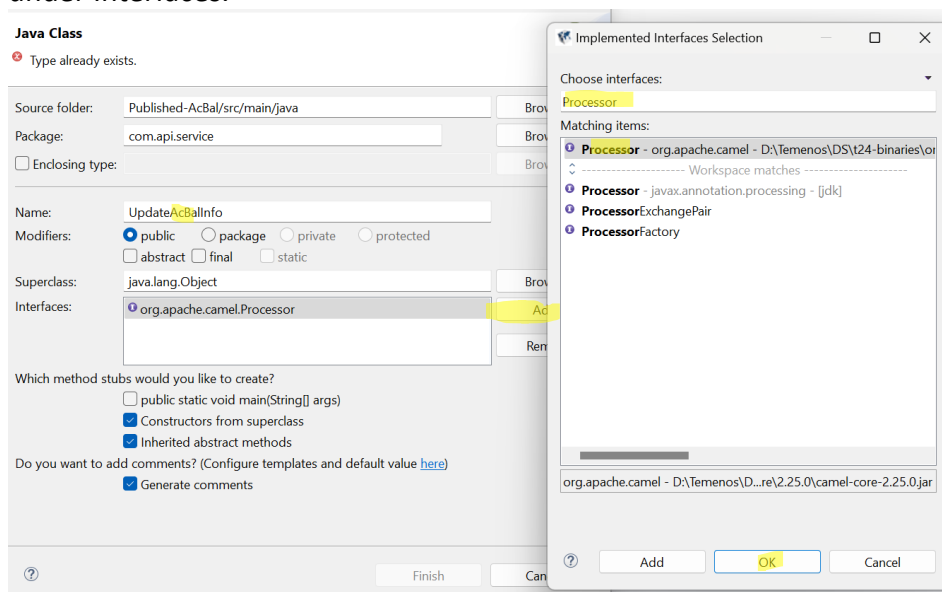
    <artifactId>threetenbp</artifactId>

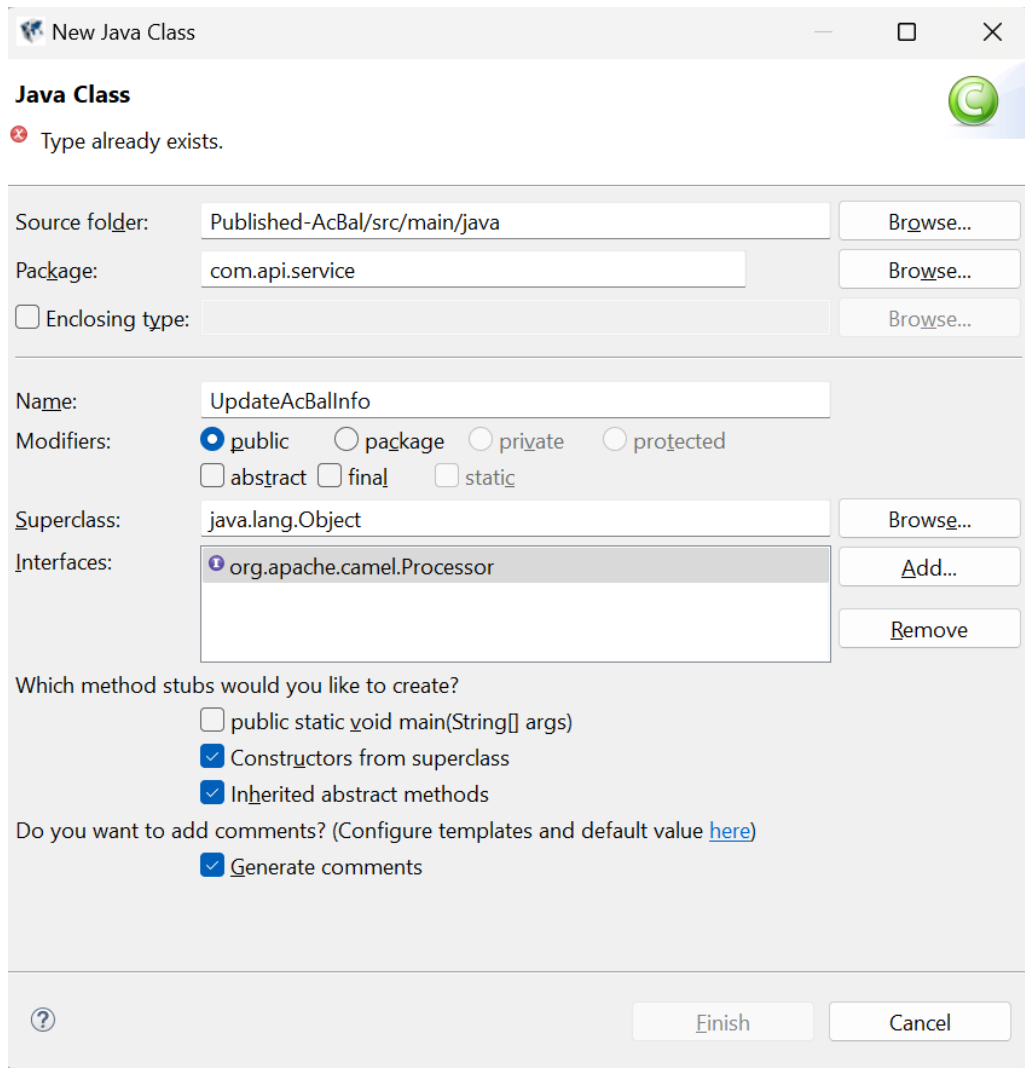
    <version>1.3.3</version>

</dependency>

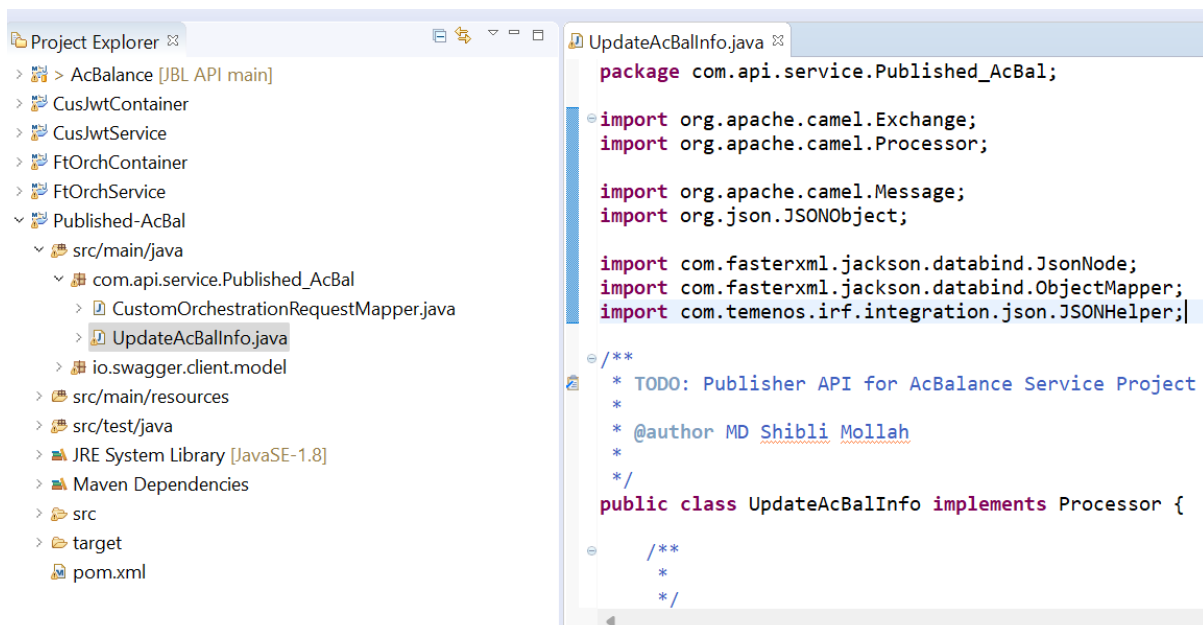
```

10. Navigate to **publisherProject - Published-AcBal**
 <<**publisherProject**>>\src\main\java\ **com.api.service**
 package directory, right click and select **New -> Java -> Class** and enter the class
 name as <<**AnyRelevantName**>>**Mapper** and add the interface named "Processor"
 under Interfaces.





11. The mapper class will be generated as shown below.



The `process()` method in the mapper must have the code logic to transform the provider API to publisher API and vice versa.

The code logic must be in such a way that the fields present in the **JSON response** body of the **provider** API are mapped to the fields defined in the swagger of **published** API using the **published** API swagger classes generated.

Mapper class for ENQUIRY based API:

For **ENQUIRY** API, we will require only a **response** mapper since we **do not send** a request body.

Below are the steps to create a Response Mapper for an ENQUIRY API based on ACCOUNT application.

a. As mentioned in **step 10**, create a **mapper class** with a relevant name (**UpdateAcBalInfo**) and Processor interface. In the generated mapper java file, **comment the constructor** (if not required). Only the **process** method will be used.

b. To map the fields of the provider API, we must first receive the header and JSON body content (containing header and body section of JSON response) of the provider API after ensuring the http response code is **200**. Below is the corresponding code line. On confirming it is a successful response, we obtain the JSON response – header and body section.

```
public void process(Exchange exchange) throws Exception {
    // TODO Auto-generated method stub
    String httpResponseCode =
exchange.getIn().getHeaders().get("CamelHttpResponseCode").toString();
    Message in = exchange.getIn();
    String response = (String) in.getBody(String.class);
    JSONObject = new JSONObject(response);
    JSONObject jsonObject = jsonObject.getJSONObject("header");
    String updateResponse = null;

    if (HttpResponseCode.equals("200")) {

        JSONObject jsonBody =
jsonObject.getJSONArray("body").getJSONObject(0);
```

Receive the header and body content of provider JSON response.

```
String responseCode = null;
    String accountNo = null;
    String alternateAccountNo = null;
    String currency = null;
    String balance = null;

    try {
        responseCode = jsonBody.get("response").toString();
        accountNo = jsonBody.get("accountId").toString();
        alternateAccountNo = jsonBody.get("accountIBAN").toString();
        currency = jsonBody.get("currency").toString();
        balance = jsonBody.get("availableBalance").toString();
```

```

    } catch (Exception e) {}

    jsonBody.remove("accountId");
    jsonBody.remove("accountIBAN");
    // jsonBody.remove("currency");
    jsonBody.remove("availableBalance");

    jsonBody.put("accountNo", accountNo);
    jsonBody.put("alternateAccountNo", alternateAccountNo);
    jsonBody.put("balance", balance);

    jsonHeader.remove("audit");
    jsonHeader.remove("page_start");
    jsonHeader.remove("page_token");
    jsonHeader.remove("total_size");
    jsonHeader.remove("page_size");
    jsonHeader.remove("status");

    jsonHeader.put("statusCode", responseCode);
    updateResponse = "{ \"header\": " + jsonHeader + ", \"body\": " + jsonBody
+ " }";
}

ObjectMapper om = JSONHelper.createObjectMapper();
JsonNode modifiedResponse = om.readTree(updateResponse);
in.setBody(modifiedResponse);
}

```

Sample Mapper Class:



UpdateAcInfo.java



UpdateAcBallInfo.java

c. Edit the **service.xml** of the publisher API by adding this **ResponseMapper** (*UpdateAcBallInfo*) as a process after redirecting it to the provider API.

```

party-v1.0.0-published-myacbal-service-v1.0.0.xml
<camelContext xmlns="http://camel.apache.org/schema/spring" id="party-v1.0.0-published">
  <onException>
    <exception>java.lang.Exception</exception>
    <handled>
      <constant>true</constant>
    </handled>
    <process ref="exceptionHandler"/>
  </onException>
  <restConfiguration component="servlet" producerApiDoc="published-myAcBal-v1.0.0-sw">
    <rest path="/v1.0.0/party/v1.0.0" produces="application/json" id="party-v1.0.0.ser">
      <get uri="/party/accounts/{accountId}" id="getAcBal">
        <param name="accountId" type="path" required="true"/>
        <!-- Publisher API Uri -->
        <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
      </get>
    </rest>
    <route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
      <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
      <!-- Provider API Uri -->
      <to uri="direct-vm:party-accounts.v1.0.0.getAcBal"/>
      <process ref="UpdateAcBallInfo"/>
    </route>
    <route id="direct.mockResponder">
      <from uri="direct:mockResponder"/>
      <process ref="mockResponder"/>
    </route>
  </camelContext>
  <bean id="UpdateAcBallInfo" class="com.api.service.UpdateAcBallInfo"/>
</beans>

```

```

<get uri="/party/accounts/{accountId}" id="getAcBal">
    <param name="accountId" type="path" required="true"/>
    <!-- Publisher API Uri -->
    <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
</get>
</rest>
<route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
    <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
    <!-- Provider API Uri -->
    <to uri="direct-vm:party-accounts.v1.0.0.getAcBal"/>
    <process ref="UpdateAcBalInfo"/>
</route>
<route id="direct.mockResponder">
    <from uri="direct:mockResponder"/>
    <process ref="mockResponder"/>
</route>

```

Add the bean:

```

<bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo"/>

```



party-v1.0.0-published-myacbal-service-v1.0.0.xml

Special Note: We must redirect the URI of the other versions of the service xml's of the Publisher API by changing like this, otherwise war file deployment will be **failed**.

```

<route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
    <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
    <!-- OWN PUBLISHER API Uri -->
    <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
</route>

```

```

party-v1.0.0-published-myacbal-service-v1.0.xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.springframework.org/schema/xsi"
    xsi:schemaLocation="
        http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd
        http://camel.apache.org/schema/spring
        http://camel.apache.org/schema/spring/camel-spring.xsd">
    <camelContext xmlns="http://camel.apache.org/schema/spring" id="party-v1.0.0-published-myAcBal.service.v1.0">
        <onException>
            <exception>java.lang.Exception</exception>
            <handled>
                <constant>true</constant>
            </handled>
            <process ref="exceptionHandler"/>
        </onException>
        <restConfiguration component="servlet" producerApiDoc="published-myAcBal-v1.0-swagger.json" bindingMode="aut">
            <rest path="/v1.0/party/v1.0.0" produces="application/json" id="party-v1.0.0.service.restLet">
                <get uri="/party/accounts/{accountId}" id="getAcBal">
                    <param name="accountId" type="path" required="true"/>
                    <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
                </get>
            </rest>
            <route id="direct-vm.party-v1.0.0.v1.0.0.getAcBal">
                <from uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
                <!-- OWN PUBLISHER API Uri -->
                <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
            </route>
        </camelContext>
    </beans>

```

```

party-v1.0.0-published-myacbal-service-v1.xml
xsi:schemaLocation="
  http://www.springframework.org/schema/beans
  http://www.springframework.org/schema/beans/spring-beans.xsd
  http://camel.apache.org/schema/spring
  http://camel.apache.org/schema/spring/camel-spring.xsd">
<camelContext xmlns="http://camel.apache.org/schema/spring" id="party-v1.0.0-published-myAcBal.service.v1">
  <onException>
    <exception>java.lang.Exception</exception>
    <handled>
      <constant>true</constant>
    </handled>
    <process ref="exceptionHandler"/>
  </onException>
  <restConfiguration component="servlet" producerApiDoc="published-myAcBal-v1-swagger.json" bindingMode="auto">
  <rest path="/v1/party/v1.0.0" produces="application/json" id="party-v1.0.0.service.restlet">
    <get uri="/party/accounts/{accountId}" id="getAcBal">
      <param name="accountId" type="path" required="true"/>
      <to uri="direct-vm:party-v1.0.0.v1.getAcBal"/>
    </get>
  </rest>
  <route id="direct-vm.party-v1.0.0.v1.getAcBal">
    <from uri="direct-vm:party-v1.0.0.v1.getAcBal"/>
    <!-- OWN PUBLISHER API Uri -->
    <to uri="direct-vm:party-v1.0.0.v1.0.0.getAcBal"/>
  </route>
</camelContext>
</beans>

```



party-v1.0.0-published-myacbal-service-v1.0.xml



party-v1.0.0-published-myacbal-service-v1.xml

11. Maven Install the **publisher API** project(s).



Published-AcBal-0.0.1-SNAPSHOT.jar

12. Include the **publisher API** and **provider API** project(s) dependency in the **container** project's pom.xml and perform maven install.

UpdateAcInfo.java AcBalPublishedContainer/pom.xml

Dependencies Overriding managed version 21.0.38 for irf-api-generator (Click for 3

Dependencies	Actions
camel-dozer (managed:2.25.0)	Add... Remove Properties... Manage...
camel-jackson (managed:2.25.0)	
junit [test] (managed:4.13.2)	
jackson-databind (managed:2.11.0)	
irf-event-request-logger : \${irf-version} (managed:21.0.38)	
irf-git-service : \${irf-version}	
irf-provider-oa : \${irf-version}	
irf-traceability-comms : \${irf-version} (managed:21.0.38)	
PSD2-response-jwt-token-generator : \${irf-version} (managed:21.0.38)	
irf-microservices-common : \${irf-version}	
irf-comms-http : \${irf-version}	
irf-pap-services : \${irf-version}	
irf-pap-model : \${irf-version}	
irf-core-system-services : \${irf-version}	
Service-AcBal-Published : 0.0.1-SNAPSHOT	
AcBalance : 0.0.1-SNAPSHOT	

```

<dependency>
  <groupId>com.api.service</groupId>
  <artifactId>Service-AcBal-Published</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
<dependency>
  <groupId>AcBal</groupId>
  <artifactId>AcBalance</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>

```

14. Create a new bean with **any relevant name** for "id" attribute in **applicationContext.xml** of the **container project** for the mapper class created. Enable Basic Auth/JWT required.

```

<!-- Response mapper added -->
  <bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo">
</bean>

```

applicationContext.xml

```

<!-- Response mapper added -->
<bean id="UpdateAcBalInfo" class="com.api.service.UpdateAcBalInfo">
</bean>

<!-- <bean id="t24SecurityFilter" class="com.temenos.irf.comms.security.defaultimpl.NullBean" /> -->

<!-- <bean id="t24SecurityFilter" class="com.temenos.irf.security.t24.T24SpringSecurityContextFilter" /> -->

<!-- Comment the above bean with id t24SecurityFilter and uncomment below bean for Security Filter -->
<!--<bean id="t24SecurityFilter" class="com.temenos.irf.comms.security.defaultimpl.T24Security" /> -->

<!-- Comment the above bean with id t24SecurityFilter and uncomment below bean for Basic Authentication -->
<bean id="t24SecurityFilter" class="com.temenos.irf.comms.security.defaultimpl.T24BasicAuthenticationCheck" />

<bean id="serviceLocatorProperties" class="com.temenos.irf.config.StandardPropertyReader">
  <property name="path" value="classpath:/irf-config/service-locator.properties" />
</bean>

<bean id="mockDataMgmtProcessor" class="com.temenos.irf.mock.MockDataMgmtProcessor">
  <!-- basePath should be set to the base Directory where mockFiles folder is placed. -->
  <property name="basePath" value="classpath:/irf-config/"></property>
</bean>

```

SAVE it.

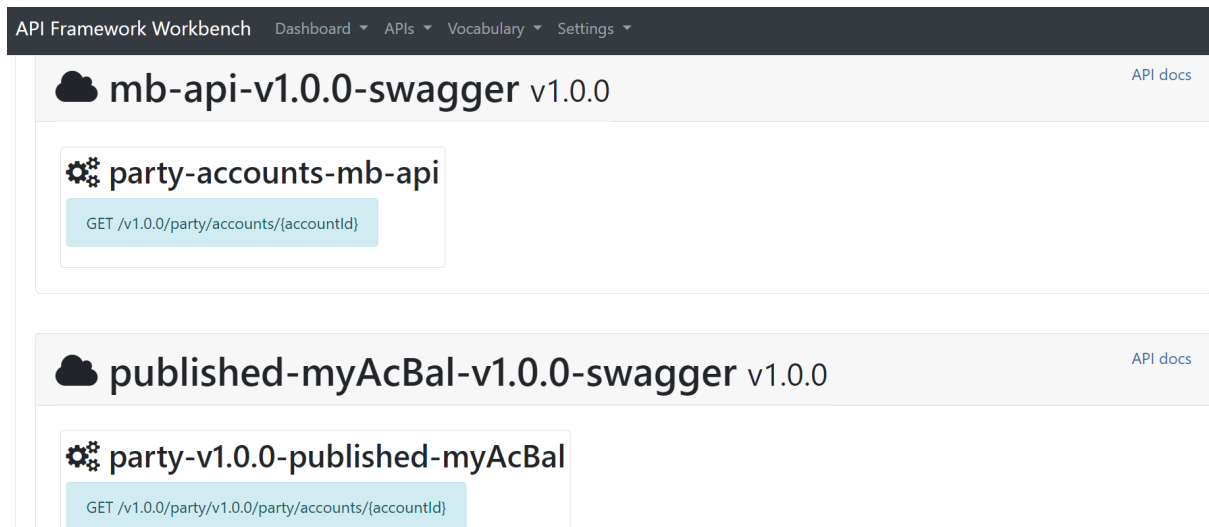


PublishedAcBalContainer.war

14. Deploy war in application server (or) jetty server.

This PC > Local Disk (D:) > Temenos > jboss > standalone > deployments > Search deployments				
<div> </div> <div> Sort View </div>				
	Name	Date modified	Type	Size
	PublishedAcBalContainer.war	18/02/2024 3:25 PM	WAR File	142,515 KB
	PublishedAcBalContainer.war.deployed	18/02/2024 3:25 PM	DEPLOYED File	1 KB

15. APIs can now be accessed with structure of published api json and the corresponding response is now formed as per requirement.



Below screenshots show the JSON requests and responses for the examples explained in this document.

