



Webservice Caller utility

July 2019

Document Control

Revision History

Date	Version	Revision	Description	Author(s)
10.July.2019	1.0	1.0	Initial Draft	Ranjith Vijayan

Contents

1. Introduction	4
2. Deployment units.....	4
2.1 J2EE layer units - ExtWSCallUtil.war.....	4
2.2 DB layer units - Ifpks_http_servlet.spc/sql and Ifpks_http_mdb.spc/sql	5
3. Configuration	5
3.1 Configurations for MDB Option	5
3.1.1 Configuration steps on DB layer.....	5
• Grant execute on required dbms packages	5
• Create AQ tables and Queues	5
• Compile database units for API.....	6
3.1.2 Configurations steps on Weblogic server	6
3.2 Configuration for Servlet Option.....	8
3.2.1 Configuration steps on J2EE layer.....	8
3.2.2 Configuration steps on DB layer.....	8
4. API specifications.....	9
5. Usages and Examples.....	10
5.1 Calling a GET URL.....	10
5.1.1 Using servlet option	10
5.1.2 Using MDB option	10
5.2 Calling a REST service of GET type.....	11
5.3 Calling a REST service of POST type	11
5.4 Calling a SOAP service of POST type.....	11
5.5 Sending additional HTTP header parameters.....	14
5.5.1 Option 1 by passing explicit type variable	14
5.5.2 Option 2 by setting session variable	14
6. Additional points	15
6.1 Logging / Debugging	15
6.2 Calling HTTPS endpoints over SSL.....	15
6.3 Using the J2EE layer without DB APIs	15
7. Annexure.....	17
7.1 References.....	17

1. Introduction

This utility can be used for making external webservice calls using J2EE layer. Database applications can use this utility for making webservice call using PL/SQL, so that establishing a direct connectivity to the external webservice endpoint is not required from Database Server. The utility comes with 2 options viz Servlet option and MDB option. Both these modes are packaged into the same war file, however typically either one of the modes will be used for implementation, and un-used mode should be disabled before deploying.

- **MDB Option**

In this mode, an MDB deployed on Weblogic server listens to a local JMS Queue which is interoperated with database's Advanced queue (AQ). Once a message is received in this queue, call to external endpoint will be made, and response will be put into a reply Queue (JMS) which is also connected to AQ. Oracle AQ (Advanced Queue) can interoperate with Weblogic JMS synchronously. Using this feature messages can be exchanged between J2EE layer to DB layer without expensive poller mechanism.

- **Servlet Option**

A servlet deployed on J2EE server can be called from PL/SQL (or any http client). The servlet will make call to the external endpoint and the response received from the external endpoint will be replayed back in the payload to caller.

Though this utility is designed primarily for making calls to external webservice from DB, the Servlet mode can be used by any http client (such as Java script from browser) and MDB mode can be used by any JMS client. This document is more focused on using this utility using PL/SQL.

Features:

- The utility supports HTTP verbs GET, POST, PUT, PATCH and DELETE. It uses Apache CloseableHttpClient API, and required external libraries are included in the war file's lib folder.
- Calls over SSL is supported, and hence calls can be made to external webservice endpoints that are using HTTPS protocol.
- This utility can be used for calling both REST and SOAP endpoints (or any HTTP endpoints for that matter). As such the utility will be unaware of the message protocol, as it's a simple mechanism to make HTTP calls. The required URI/HTTP Headers/Verb (GET or POST) and payload (in case of POST) will have to be supplied to the utility and it will just make HTTP calls to the requested URI. A PL/SQL API is provided for making calls to this utility.

2. Deployment units

The utility essentially has deployable units on J2EE layer and DB layer

2.1 J2EE layer units - ExtWSCallUtil.war

This war file can be deployed on any J2EE server. For using MDB mode with interoperability with database AQ, it is assumed to be deployed on a Weblogic server.

Both the servlet and MDB are packaged in this war file. If you want to use only one of the modes, the other mode should ideally be disabled by removing the inappropriate deployment descriptor files.

To disable MDB mode – just remove ejb-jar.xml and weblogic-ejab-jar.xml from WEB-INF folder, and to disable Servlet mode, remove web.xml and weblogic.xml from WEB-INF folder.

Servlet is protected with Basic authentication scheme and hence a user id and password need to be sent for consuming the servlet. If the file is deployed on weblogic server, this will be a valid weblogic user id and password

2.2 DB layer units - Ifpks_http_servlet.spc/sql and Ifpks_http_mdb.spc/sql

These units contain the PL/SQL APIs for making external webservices calls from DB via the J2EE layer.

For MDB mode, APIs are given in ifpks_http_mdb. It requires 2 AQs to be created as described in the config section, and the names of these should be updated in the package variables g_mdb_req_q and g_mdb_reply_q. (Ideally these would be parameterized in some tables and initialized. One needs to customize this part)

For servlet mode, APIs are given in ifpks_http_servlet. It requires the servlet URL and servlet user and password provided in the variables g_servlet_url, g_servlet_uid, g_servlet_pwd.

3. Configuration

Please follow the instructions in the required section of this chapter, depending on whether you want to use MDB option or Servlet option

3.1 Configurations for MDB Option

3.1.1 Configuration steps on DB layer

- **Grant execute on required dbms packages**

Using database SYS user grant execute on dbms_aq to the database schema

```
grant execute on dbms_aq to userschema;  
grant execute on dbms_aqadm to userschema;  
grant execute on dbms_aqin to userschema;
```

- **Create AQ tables and Queues**

Connect to database user schema and Create AQ queue tables and queues. 2 AQ queues are required to be created – one for putting request and one for getting response. Below stub can be used to create queues. The names used in the below stub (Q_QT,RQ,RQT) can be changed appropriately, and it needs to be used while creating the JMS, and supply the same in ifpks_http_mdb.g_mdb_req_q and g_mdb_resp_q accordingly.

```

begin
-- Request queue
dbms_aqadm.create_queue_table('QT', 'SYS.AQ$_JMS_TEXT_MESSAGE');
dbms_aqadm.create_queue('Q', 'QT');
dbms_aqadm.start_queue('Q');

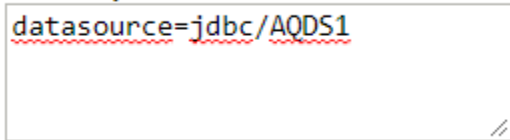
-- Reply queue
dbms_aqadm.create_queue_table('RQT', 'SYS.AQ$_JMS_TEXT_MESSAGE');
dbms_aqadm.create_queue('RQ', 'RQT');
dbms_aqadm.start_queue('RQ');
end;
/

```

- **Compile database units for API**

Compile the database unit ifpks_http_mdb spc/sql. The AQ names used in the above step1, can be updated in the package spec, or customize it to supply from a config table.

3.1.2 Configurations steps on Weblogic server

- Login to Weblogic server, and follow the steps mentioned in document **AQ_JMS_Queue_Configuration_steps.pdf**. Detailed steps with screenshots are given in this document to create below configurations. Important points are noted below for reference
 - **Create Datasource** (Services->Datasources)
 - Notes:*
 - XA Datasource should be used
 - For the property “**Maximum Capacity**” give a sufficiently higher value like “150” than the default 10.
 - The JNDI Name of the datasource will be used this in the next step while configuring JMS foreign server.
 - **Create JMS module for AQ JMS** (Services->Messaging->JMS Modules)
 - **Create JMS Foreign Server** (Inside the above JMS Modules, create a New Foreign server)
 - Notes:*
 - JNDI Initial Context Factory:** Should be `oracle.jms.AQjmsInitialContextFactory`
 - in **JNDI Properties:** text box please provide a string “datasource=<JNDI Name of the data source created in above step>”
E.g.
JNDI Properties:

 - **Create JMS Foreign Connection Factory** (Inside the above Foreign Server, go to the Connection Factories tab)

Notes:

- v. **Remote JNDI Name:** Should be **XAQueueConnectionFactory**
- vi. Note down the Local JNDI name, as it will be required in the MDB's deployment descriptor files
- o **Create JMS Destination Queues** (Inside the above Foreign Server, go to the Destinations tab)

Two queues are required- one for Request, binding to remote AQ queue Queues/Q and one for Reply queue binding to AQ Queues/RQ

Notes:

- vii. **Remote JNDI Name:** should be "Queues/" followed by the Database AQ Name
E.g. Queues/**Q** for request Queue and Queues/**RQ** for response queue as per the names used in this document.
- viii. Note down the Local JNDI names, as it will be required in the MDB's deployment descriptor files

Foreign Destinations

New Delete			
<input type="checkbox"/>	Name ↕	Local JNDI Name	Remote JNDI Name
<input type="checkbox"/>	replyQ	replyQ	Queues/RQ
<input type="checkbox"/>	requestQ	requestQ	Queues/Q
New Delete			

- Update the war file's deployment descriptor files with the JNDI Names of Queue Connection Factory and Destination Queues created in the above steps.
 - o In the file WEB-INF/weblogic-ejbjar.xml
 - Update **connection-factory-jndi-name** tag with value as the Local JNDI name of Connection Factory created in the above steps
 - Update **destination-jndi-name** tag with value as the Local JNDI name of Request Queue created in the above steps
 - o In the file WEB-INF/ejb-jar.xml and
 - In the node **env-entry** with **env-entry-name** tag **ReplyQueue**, update **env-entry-value** tag with value as the JNDI name of the Reply queue created in above steps
 - In the node **env-entry** with **env-entry-name** tag **ReplyQueueCF**, update **env-entry-value** tag with value as the JNDI name of Connection Factory created in the above steps
- Deploy the war file ExtWSCallUtil.war on weblogic server as a normal application deployment. Optionally you may remove the files WEB-INF/web.xml and WEB-INF/weblogic.xml to disable servlet mode. Restart the server if required, and now the MDB will start listening on the Database AQ

3.2 Configuration for Servlet Option

3.2.1 Configuration steps on J2EE layer

Deploy the war file ExtWSCallUtil.war on weblogic server (or any J2EE server) as a normal application deployment. Optionally you may remove the files WEB-INF/ejb-jar.xml and WEB-INF/weblogic-ejb-jar.xml to disable the MDB mode.

The servlet URL will be `http(s)://<host>:<port>/WSCallerServlet/WSCallerServlet`.

3.2.2 Configuration steps on DB layer

- Connect to sys account in database and run below stub. Change the highlighted text with appropriate values

```
grant execute on utl_http to userschema;
begin
dbms_network_acl_admin.create_acl (
acl          => 'utlhttp.xml',
description  => 'acl description',
principal    => 'userschema',
is_grant     => true,
privilege    => 'connect',
start_date   => null,
end_date     => null);
commit;
end;
/
begin
dbms_network_acl_admin.add_privilege (
acl          => 'utlhttp.xml',
principal    => 'userschema',
is_grant     => false,
privilege    => 'connect',
position     => null,
start_date   => null,
end_date     => null);
commit;
end;
/
```



```

begin
dbms_network_acl_admin.assign_acl(
acl      => 'utlhttp.xml',
host      => 'host/ip of the servlet host',
lower_port => 'port of the servlet',
upper_port => 'port of the servlet');
commit;
end;
/

```

- If the servlet URL is on HTTPS mode, you will have to configure the oracle wallet using orapki or wallet manager. Refer the document “**Calling Webservices using PLSQL.pdf**” for the configuration of this.
- Compile the database units ifpks_http_servlet in the database scheme. The servlet URL and weblogic credentials needs to be updated in this unit, or supply it at runtime

4. API specifications

The packages `ifpks_http_servlet` and `ifpks_http_mdb` has below methods for making the http calls. All these methods will return a clob response with payload received from the actual webservice endpoint.

Method	Type	Purpose
<code>fn_http_get</code>	function	for making HTTP GET calls
<code>fn_http_post</code>	function	for making HTTP POST calls
<code>fn_http_put</code>	function	for making HTTP PUT calls
<code>fn_http_patch</code>	function	for making HTTP PATCH calls
<code>fn_http_delete</code>	function	for making HTTP DELETE calls

Following are the arguments to be passed to above methods

Argument	Type	Remarks
<code>p_url</code>	Varchar2	URL of the webservice endpoint. It can be HTTP or HTTPS URL. The J2EE component will make the HTTP connection to this end point
<code>p_payload</code>	Clob	Payload that needs to be sent to the webservice. This is applicable only for POST/PUT/PATCH methods, and not for GET/DELETE methods
<code>p_content_type</code>	Varchar2	Optional parameter only for POST/PUT/PATCH methods. - not applicable for GET/DELETE methods.

		If this is sent, "Content-Type" HTTP header will be created. Some examples of valid content type are <ul style="list-style-type: none"> • text/xml • application/json • application/x-www-form-urlencoded
p_header	Type	Optional parameter for additional HTTP headers that needs to be set while making the HTTP call. If this parameter is not sent to the API, then default values set in the package session variable tb_header will be taken

5. Usages and Examples

Examples in this section are for calling the webservice using the PL/SQL APIs ifpks_http_servlet or ifpks_http_mdb. Both these APIs offer same functionality.

5.1 Calling a GET URL

5.1.1 Using servlet option

```
SQL> set line 9999 pages 100 long 999999999

SQL> select ifpks_http_servlet.fn_http_get(p_url=>'https://www.google.com') from dual;

IFPKS_HTTP_SERVLET.FN_HTTP_GET(P_URL=>'HTTPS://WWW.GOOGLE.COM')
```

```
-----

!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-P
H"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><met
a content="/images/branding/google/1x/google_standard_color_128dp.png" itempro
p="image"><title>Google</title><script nonce="rusI0y9avuid5I4PGHaxOw==">(functio
n(){window.google={kEI:'wC8nXbnlBYLlwAPrtJigCQ',kEXPI:'0,1353804,1957,1641,782,1
225,730,224,510,1065,2081,1071,57,320,207,491,526,1203,79,202,33,23,184,55,330,1
14,2331399,329502,1294,12383,4855,32692,15247,867,12163,6381,853,2482,2,2,6801,3
69,3314,5505,224,2212,266,5107,575,835,284,2,579,727,2432,58,2,4,1297,4323,4968,
773,2247,1410,1414,3069,9,1968,2590,3601,669,1050,1808,1397,81,7,3,488,2044,8909
,4604,693,796,1220,661,295,756,119,38,1179,1364,1611,2692,44,3061,2,631,2403,837
,44,2376,2407,2607,12,622,2226,655,21,317,1118,448,7,447,190,2,961,198,777,1,371
,1314,705,756,98,392,30,399,472,520,1107,10,168,8,109,1018,1495,174,889,78,48,55
3,11,14,10,1269,2212,25,177,323,5,62,1183,7,840,324,193,531,262,691,38,158,2126,
---snipped---
```

5.1.2 Using MDB option

```
SQL> select ifpks_http_mdb.fn_http_get(p_url=>'https://www.google.com') from dual;

IFPKS_HTTP_MDB.FN_HTTP_GET(P_URL=>'HTTPS://WWW.GOOGLE.COM')
```

```
-----

!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="en-P
H"><head><meta content="text/html; charset=UTF-8" http-equiv="Content-Type"><met
```

```
a content="/images/branding/googleg/1x/googleg_standard_color_128dp.png" itempro
p="image"><title>Google</title><script nonce="QegFXKbn5tTp3MRqN8Eq1g==">(functio
n(){window.google={kEI:'hS8nXcGZINmg-Qa3gJnoCg',kEXPI:'0,18167,1335637,1957,2423
,1225,730,224,510,18,227,820,3151,379,206,1017,175,1032,75,202,239,56,329,78,10,
1133518,1197917,303188,26305,1294,12383,4855,32691,15248,867,12163,6381,3335,2,2
,6801,369,3314,5505,224,2212,266,5107,575,835,284,2,1306,2431,59,2,4,1297,4323,4
968,773,2251,1406,4483,9,981,3577,1929,1672,669,1050,1808,1397,81,7,3,472,16,204
4,8909,1902,3395,796,101,777,342,38,920,754,119,1217,1364,1611,2736,3061,2,632,3
239,44,4783,2607,12,620,803,336,1089,656,20,318,1118,474,7,423,187,2,961,198,777
,1,368,1317,80,625,757,97,392,29,400,992,1107,10,168,8,109,1018,89,1406,174,814,
153,48,553,11,14,10,657,612,1436,776,25,177,323,5,62,1183,7,842,322,193,14,499,1
---snipped---
```

5.2 Calling a REST service of GET type

```
SQL> select ifpks_http_mdb.fn_http_get(p_url => 'https://reqres.in/api/users?page=1')
from dual;
```

```
IFPKS_HTTP_MDB.FN_HTTP_GET(P_URL=>'HTTPS://REQRES.IN/API/USERS?PAGE=1')
```

```
-----
"page":1,"per_page":3,"total":12,"total_pages":4,"data":[{"id":1,"email":"george
.bluth@reqres.in","first_name":"George","last_name":"Bluth","avatar":"https://s3
.amazonaws.com/uifaces/faces/twitter/calebogden/128.jpg"}, {"id":2,"email":"janet
.weaver@reqres.in","first_name":"Janet","last_name":"Weaver","avatar":"https://s
3.amazonaws.com/uifaces/faces/twitter/josephstein/128.jpg"}, {"id":3,"email":"emm
a.wong@reqres.in","first_name":"Emma","last_name":"Wong","avatar":"https://s3.am
azonaws.com/uifaces/faces/twitter/olegpogodaev/128.jpg"}]}
```

5.3 Calling a REST service of POST type

```
SQL> select ifpks_http_mdb.fn_http_post
2      (
3      p_url => 'https://reqres.in/api/users',
4      p_content_type => 'application/json',
5      p_payload=>'{
6          "name": "Ranjith",
7          "job": "Artist"}'
8      )
9 from dual;
```

```
IFPKS_HTTP_MDB.FN_HTTP_POST(P_URL=>'HTTPS://REQRES.IN/API/USERS',P_CONTENT_TYPE=
-----
```

```
"name":"Ranjith","job":"Artist","id":"863","createdAt":"2019-07-11T12:52:22.138Z
"}
}
```

5.4 Calling a SOAP service of POST type

```
SQL> 1
1 select ifpks_http_mdb.fn_http_post
2      (
3
4      p_url=>'http://210.61.14.53:17005/FCUBSCustomerService/FCUBSCustomerService',
5      p_content_type => 'text/xml',
6      p_payload=>
```

```

6  '<soapenv:Envelope
7    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
8    <soapenv:Header/>
9    <soapenv:Body>
10      <CREATECUSTOMER_FSFS_REQ
11        xmlns="http://fcubs.ofss.com/service/FCUBSCustomerService">
12        <FCUBS_HEADER>
13          <SOURCE>FCAT</SOURCE>
14          <UBSCOMP>FCUBS</UBSCOMP>
15          <USERID>FCATOP</USERID>
16          <BRANCH>074</BRANCH>
17          <SERVICE>FCUBSCustomerService</SERVICE>
18          <OPERATION>CreateCustomer</OPERATION>
19        </FCUBS_HEADER>
20        <FCUBS_BODY>
21          <Customer-Full>
22            <PRIVATE_CUSTOMER>N</PRIVATE_CUSTOMER>
23            <CTYPE>I</CTYPE>
24            <NAME>RANJITH439</NAME>
25            <SNAME>US-TEST2</SNAME>
26            <ADDRLN1>TAIWAN MAIN STREET</ADDRLN1>
27            <ADDRLN2>MARKET CORNER</ADDRLN2>
28            <COUNTRY>TW</COUNTRY>
29            <NLTY>US</NLTY>
30            <LBRN>074</LBRN>
31            <CCATEG>INDIVIDUAL</CCATEG>
32            <FULLNAME>YAN TING</FULLNAME>
33            <MEDIA>MAIL</MEDIA>
34            <LOC>OT</LOC>
35            <CREATEACC>Y</CREATEACC>
36            <CREATEDACCOUNT>Y</CREATEDACCOUNT>
37            <UIDVAL>PASSPORTTEST1</UIDVAL>
38            <UIDNAME>PASSPORT NUMBER</UIDNAME>
39            <Cust-Account>
40              <BRN>074</BRN>
41              <CCY>TWD</CCY>
42              <ACCOUNT_CLASS>CAOFF1</ACCOUNT_CLASS>
43              <AC_OPEN_DATE>2018-07-02</AC_OPEN_DATE>
44              <CHKNAME1>123</CHKNAME1>
45              <LOCATION>NY</LOCATION>
46            </Cust-Account>
47            <Custpersonal>
48              <DOB>1974-05-15</DOB>
49              <GENDR>M</GENDR>
50              <LANG>CHS</LANG>
51              <BIRTHCOUNTRY>TW</BIRTHCOUNTRY>
52            </Custpersonal>
53            <Cust-Add1>
54              <AML_CFT_CONCERN_IND>Y</AML_CFT_CONCERN_IND>
55              <BANKCODEOFDEPOSITACC>004</BANKCODEOFDEPOSITACC>
56              <CELPHONE>589890903</CELPHONE>
57              <COMPANYNAME>Seri</COMPANYNAME>
58              <CRSDECLARATION>Y</CRSDECLARATION>
59              <CUSTONBOARDPURPOSE>1</CUSTONBOARDPURPOSE>
60              <DEPOSITACCOTHBANK>6579900954</DEPOSITACCOTHBANK>
61              <ENGLISH_NAME>Seri</ENGLISH_NAME>
62              <FATCADECLARATION>N</FATCADECLARATION>
63              <JOB_TITLE>Prime Minister</JOB_TITLE>
64              <NATID_ISSUEDATE>2019-05-02</NATID_ISSUEDATE>
65              <NATID_ISSUEPLACE>Taipei</NATID_ISSUEPLACE>
66              <NATID_ISSUESTATUS>1</NATID_ISSUESTATUS>
67              <NEGATIVE_NEWS_IND>N</NEGATIVE_NEWS_IND>
68              <OCCUPATION_CODE>PM</OCCUPATION_CODE>

```

```

69          <OFFICEPHONE>589890903</OFFICEPHONE>
70          <ONBOARDING_CHANNEL>1</ONBOARDING_CHANNEL>
71          <PANAMA_IND>Y</PANAMA_IND>
72
<POLITICALLY_EXPOSED_PERSON_IND>N</POLITICALLY_EXPOSED_PERSON_IND>
73          <PRODUCTS_TYPE>1</PRODUCTS_TYPE>
74          <RESTRICTED_CUSTOMER>N</RESTRICTED_CUSTOMER>
75          <SANCTIONED_CUSTOMER>N</SANCTIONED_CUSTOMER>
76          <SECIDENTITYNO>999</SECIDENTITYNO>
77          <SECIDENTITYTYPE>1</SECIDENTITYTYPE>
78
<FOREIGNER_LONGSTAY_INDICATOR>Y</FOREIGNER_LONGSTAY_INDICATOR>
79          </Cust-Add1>
80          </Customer-Full>
81          </FCUBS_BODY>
82          </CREATECUSTOMER_FSFS_REQ>
83          </soapenv:Body>
84          </soapenv:Envelope>')
85* from dual
SQL> /

```

```

IFPKS_HTTP_MDB.FN_HTTP_POST(P_URL=>'HTTP://210.61.14.53:17005/FCUBSCUSTOMERSERVI
-----

```

```

?xml version='1.0' encoding='UTF-8'?><S:Envelope xmlns:S="http://schemas.xmlsoap
.org/soap/envelope/"><S:Body><CREATECUSTOMER_FSFS_RES xmlns="http://fcubs.ofss.c
om/service/FCUBSCustomerService"><FCUBS_HEADER><SOURCE>FCAT</SOURCE><UBSCOMP>FCU
BS</UBSCOMP><MSGID>6119192000243870</MSGID><CORRELID>null</CORRELID><USERID>FCAT
OP</USERID><ENTITY>null</ENTITY><BRANCH>074</BRANCH><MODULEID>ST</MODULEID><SERV
ICE>FCUBSCustomerService</SERVICE><OPERATION>CreateCustomer</OPERATION><DESTINAT
ION>FCAT</DESTINATION><FUNCTIONID>STDCIF</FUNCTIONID><ACTION>NEW</ACTION><MSGSTA
T>FAILURE</MSGSTAT></FCUBS_HEADER><FCUBS_BODY><Customer-Full><PRIVATE_CUSTOMER>N
</PRIVATE_CUSTOMER><CTYPE>I</CTYPE><NAME>RANJITH439</NAME><ADDRLN1>TAIWAN MAIN S
TREET</ADDRLN1><ADDRLN2>MARKET CORNER</ADDRLN2><COUNTRY>TW</COUNTRY><SNAME>US-TE
ST2</SNAME><NLTY>US</NLTY><LBRN>074</LBRN><CCATEG>INDIVIDUAL</CCATEG><FULLNAME>Y
AN TING</FULLNAME><UIDNAME>PASSPORT NUMBER</UIDNAME><UIDVAL>PASSPORTTEST1</UIDVA
L><MEDIA>MAIL</MEDIA><LOC>OT</LOC><CREATEACC>Y</CREATEACC><CREATEDACCOUNT>Y</CRE
ATEDACCOUNT><Custpersonal><DOB>1974-05-15</DOB><GENDR>M</GENDR><LANG>CHS</LANG><
BIRTHCOUNTRY>TW</BIRTHCOUNTRY></Custpersonal><Cust-Add1><AML_CFT_CONCERN_IND>Y</
AML_CFT_CONCERN_IND><BANKCODEOFDEPOSITACC>004</BANKCODEOFDEPOSITACC><CELLPHONE>58
9890903</CELLPHONE><COMPANYNAME>Seri</COMPANYNAME><CRSDECLARATION>Y</CRSDECLARATI
ON><CUSTONBOARDPURPOSE>1</CUSTONBOARDPURPOSE><DEPOSITACCOTHBANK>6579900954</DEPO
SITACCOTHBANK><ENGLISH_NAME>Seri</ENGLISH_NAME><FATCADECLARATION>N</FATCADECLARA
TION><JOB_TITLE>Prime Minister</JOB_TITLE><NATID_ISSUEDATE>2019-05-02</NATID_ISS
UEDATE><NATID_ISSUEPLACE>Taipei</NATID_ISSUEPLACE><NATID_ISSUESTATUS>1</NATID_IS
SUESTATUS><NEGATIVE_NEWS_IND>N</NEGATIVE_NEWS_IND><OCCUPATION_CODE>PM</OCCUPATIO
N_CODE><OFFICEPHONE>589890903</OFFICEPHONE><PANAMA_IND>Y</PANAMA_IND><ONBOARDING
_CHANNEL>1</ONBOARDING_CHANNEL><POLITICALLY_EXPOSED_PERSON_IND>N</POLITICALLY_EX
POSED_PERSON_IND><PRODUCTS_TYPE>1</PRODUCTS_TYPE><RESTRICTED_CUSTOMER>N</RESTRIC
TED_CUSTOMER><SANCTIONED_CUSTOMER>N</SANCTIONED_CUSTOMER><SECIDENTITYNO>999</SEC
IDENTITYNO><SECIDENTITYTYPE>1</SECIDENTITYTYPE><FOREIGNER_LONGSTAY_INDICATOR>Y</
FOREIGNER_LONGSTAY_INDICATOR></Cust-Add1><Cust-Account><BRN>074</BRN><CCY>TWD</C
CY><ACCOUNT_CLASS>CAOFF1</ACCOUNT_CLASS><AC_OPEN_DATE>2018-07-02</AC_OPEN_DATE><
CHKNAME1>123</CHKNAME1><LOCATION>NY</LOCATION></Cust-Account></Customer-Full><FC
UBS_ERROR_RESP><ERROR><ECODE>ST-CIF02</ECODE><EDESC>This Short Name is already u
sed</EDESC></ERROR><ERROR><ECODE>ST-CIF24</ECODE><EDESC>Customer Unique Identifi
er Value and Name Combination does not make it unique.This Combination is alread
y being used for the Customer Number 000004167
</EDESC></ERROR><ERROR><ECODE>ST-SAVE-004</ECODE><EDESC>Failed to Save the Recor
d</EDESC></ERROR></FCUBS_ERROR_RESP></FCUBS_BODY></CREATECUSTOMER_FSFS_RES></S:B
ody></S:Envelope>

```

SQL>

5.5 Sending additional HTTP header parameters

In this example we are setting an additional http header "Authorization" with a value.

5.5.1 Option 1 by passing explicit type variable

In this example we are declaring a local variable and populate it with header values, and pass it explicitly while making the API call

```
SQL> set serverout on size 999999
SQL> 1
  1 declare
  2   l_tb_header ifpks_http_mdb.ty_header;
  3 begin
  4   l_tb_header('Authorization') := 'Basic d2VibG9naWM6T3JhY2xlMTIz';
  5   dbms_output.put_line
  6   (
  7     ifpks_http_mdb.fn_http_get
  8     (
  9       p_url=>'http://localhost:7001/ords/FCUBS_REST/RestTest/RestTest',
 10       p_header => l_tb_header
 11     )
 12   );
 13* end;
SQL> /
```

```
"items":[{"question":"Are we Ready?","answer":"Yes We
Are!!"}], "hasMore":false, "limit":0, "offset":0, "count":1, "links":[{"rel":"self", "href":
"http://localhost:7001/ords/FCUBS_REST/RestTest/RestTest/"}, {"rel":"describedby", "href
":"http://localhost:7001/ords/FCUBS_REST/metadata-catalog/RestTest/RestTest/"}]}
```

PL/SQL procedure successfully completed.

5.5.2 Option 2 by setting session variable

In this example we are populating package session header variable and making the API call without explicitly passing during http call method.

```
SQL> exec ifpks_http_mdb.tb_header('Authorization') := 'Basic
d2VibG9naWM6T3JhY2xlMTIz';
```

PL/SQL procedure successfully completed.

```
SQL> select
ifpks_http_mdb.fn_http_get('http://localhost:7001/ords/FCUBS_REST/RestTest/RestTest')
from dual;
```

```
IFPKS_HTTP_MDB.FN_HTTP_GET('HTTP://LOCALHOST:7001/ORDS/FCUBS_REST/RESTTEST/RESTT
-----
```

```
"items":[{"question":"Are we Ready?","answer":"Yes We Are!!"}], "hasMore":false, "
limit":0, "offset":0, "count":1, "links":[{"rel":"self", "href":"http://localhost:70
01/ords/FCUBS_REST/RestTest/RestTest/"}, {"rel":"describedby", "href":"http://loca
lhost:7001/ords/FCUBS_REST/metadata-catalog/RestTest/RestTest/"}]}
```

6. Additional points

6.1 Logging / Debugging

Log / debug of J2EE layer is written into weblogic server log. To enable/disable logging or control log level, set the logging level at Weblogic Server.

Environment->Servers->(choose server)->Logging tab-> Advanced section

Set **Minimum severity to log:** to “Debug” and **Severity level:** to “Debug”

6.2 Calling HTTPS endpoints over SSL

Calls over SSL is supported, and hence calls can be made to external webservice endpoints that are using HTTPS protocol.

By default Java’s cacerts keystore file contains the default trusted Certificate Authority (CA) certificates, and no additional configuration will be required to make HTTPS calls if the endpoints SSL certificate is signed by trusted CAs. However for self signed certificates, the certificate needs to be imported using java keytool

E.g.

```
keytool -importcert -trustcacerts -alias localhost -file /path/to/yourcert.cer -keystore /path/to/cacerts -storepass changeit
```

6.3 Using the J2EE layer without DB APIs

As webservice caller component is running on J2EE server, it can be consumed by any HTTP client (like Javascript from browser) or JMS client. For consuming the servlet, send an HTTP post request with a payload having XML message in below format, the response received from the requested URL will be sent back in the response payload. For MDB, the message format is the same, the message would be put to the JMS Queue where the MDB is listening and the response will be given in response queue with correlation ID same as the request’s message ID.

```
<WS_REQUEST>
  <WS_HTTP_VERB>POST</WS_HTTP_VERB>
  <WS_URL>https://reqres.in/api/users/</WS_URL>
  <WS_HTTP_HEADERS>
    <WS_HTTP_HEADER>
      <WS_HTTP_HEADER_NAME>Content-Type</WS_HTTP_HEADER_NAME>
      <WS_HTTP_HEADER_VALUE>application/json</WS_HTTP_HEADER_VALUE>
    </WS_HTTP_HEADER>
    <WS_HTTP_HEADER>
      <WS_HTTP_HEADER_NAME>Accept</WS_HTTP_HEADER_NAME>
      <WS_HTTP_HEADER_VALUE>application/json</WS_HTTP_HEADER_VALUE>
    </WS_HTTP_HEADER>
  </WS_HTTP_HEADERS>
  <WS_PAYLOAD><![CDATA[
```

```
{  
  "name": "Ranjith",  
  "job": "Artist"  
}  
]]></WS_PAYLOAD>  
</WS_REQUEST>
```

E.g.

Below is the servlet request made using postman.

The screenshot shows a Postman REST client interface. The top bar indicates a POST request to the URL `http://localhost:7001/WSCallerServlet/WSCallerServlet`. The request body is a SOAP envelope, displayed in a syntax-highlighted text editor. The envelope includes headers for Content-Type and Accept, both set to `application/json`. The payload is a JSON object with `"name": "Ranjith"` and `"job": "Artist"`. The bottom section shows the response status as `200 C` (OK). The response body is displayed in a syntax-highlighted text editor, showing a JSON object with `"name": "Ranjith"`, `"job": "Artist"`, `"id": "123"`, and `"createdAt": "2019-07-12T03:08:52.700Z"`.

```
1 <WS_REQUEST>  
2   <WS_HTTP_VERB>POST</WS_HTTP_VERB>  
3   <WS_URL>https://reqres.in/api/users/</WS_URL>  
4   <WS_HTTP_HEADERS>  
5     <WS_HTTP_HEADER>  
6       <WS_HTTP_HEADER_NAME>Content-Type</WS_HTTP_HEADER_NAME>  
7       <WS_HTTP_HEADER_VALUE>application/json</WS_HTTP_HEADER_VALUE>  
8     </WS_HTTP_HEADER>  
9     <WS_HTTP_HEADER>  
10      <WS_HTTP_HEADER_NAME>Accept</WS_HTTP_HEADER_NAME>  
11      <WS_HTTP_HEADER_VALUE>application/json</WS_HTTP_HEADER_VALUE>  
12    </WS_HTTP_HEADER>  
13  </WS_HTTP_HEADERS>  
14  <WS_PAYLOAD><![CDATA[  
15  {  
16    "name": "Ranjith",  
17    "job": "Artist"  
18  }  
19  ]]></WS_PAYLOAD>  
20 </WS_REQUEST>
```

Body Cookies Headers (5) Test Results Status: 200 C

Pretty Raw Preview HTML ↻

```
i 1 "name": "Ranjith", "job": "Artist", "id": "123", "createdAt": "2019-07-12T03:08:52.700Z"  
2
```


7. Annexure

7.1 References

#	URL / attachment	Remarks
1	https://docs.oracle.com/middleware/1221/wls/JMSAD/aq_jms.htm	Interoperating with Oracle AQ JMS
2	https://www-us.apache.org/dist/httpcomponents/httpclient/binary/httpcomponents-client-4.5.6-bin.tar.gz	Library files used for Apache HTTP components
3	AQ_JMS_Queue_Configuration_steps.pdf	AD JMS Queue configuration steps
4	https://stbeehive.oracle.com/content/dam/st/JAPAC%20-%20South%20Asia%20-%20Consulting%20Documents%20Repository/Documents/19.0_RTB-Managed%20Services%20+%20Tools/15.0_Tools/Calling%20Webservices%20using%20PLSQL.pdf	Steps for Oracle wallet configuration
5		

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax: +91 22 6718 3001

www.oracle.com/financialservices/

Copyright © [2017] Oracle and/or its affiliates. All rights reserved.



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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. 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 failsafe, 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.

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 recompilations 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.

This software or hardware and documentation may provide access to or information on 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. 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.
