IBM z/OS Connect EE V3.0

Developing IMS/TM API Requesters Applications

Wildfire
Technical Hands-On Workshops

IBM

IBM Z
Wildfire Team –
Washington System Center

Lab Version Date: November 5, 2020

Table of Contents

Mitch Johnson (mitchj@us.ibm.com)

General Exercise Information and Guidelines

- ✓ The Windows artifacts for this exercise are in directory *c:\z\apiRequester*. Open a DOS command prompt Window and go to this directory using a change directory command, e.g. ***cd \z\apiRequester***

- ✓ Viewing or changing the contents of a file can easily be done by opening the Windows Explorer and going to directory *c:\z\apiRequester*. One this folder is display you can select a file and right mouse button click and select the *Open with EditPad Lite* option to open a file for viewing.

- ✓ This exercise requires using TSO user *USER1* and the TSO password for this user will be provided by the lab instructor.

- ✓ This exercise primarily uses data sets *USER1.ZCEE.CNTL* and *USER.ZCEE.SOURCE.*

- ✓ Please note that there may be minor differences between the screen shots in this exercise versus what you see when performing this exercise. These differences should not impact the completion of this exercise.

- ✓ Any time you have any questions about the use of IBM z/OS Explorer, 3270 screens, features or tools do not hesitate to ask the instructor for assistance.


Important: The REST API provider applications used in this exercise access a CICS application. The target API used in this exercise was developed using instructions provided in another exercise in this workshop. The target API could have just as easily have been anywhere in the cloud on any remote system so long as the Swagger document for the target API was available.

Overview

These exercises demonstrate the steps required to enable COBOL programs to invoke RESTful APIs. There are The API provider accesses a CICS program. Although this API is hosted in z/OS Connect EE server, these same steps could be followed to access APIs hosted anywhere in the cloud.

The process starts with the Swagger document that describes the API which the COBOL program will invoke. The Swagger document is used by the z/OS Connect EE build toolkit to (1) generate the COBOL copy books included in the COBOL program to invoke the API (2) as well as an API requester archive file (ARA) which is deployed to a z/OS Connect EE server.

The diagram shows this process at a high level. The copy books generated by the z/OS Connect EE build toolkit will be integrated into a COBOL program to provide (1) the COBOL representation of the request and response messages and to identity (2) the HTTP method to be used in the RESTful request. The API requester archive file will be deployed to a z/OS Connect EE server and used to convert (1) the COBOL request and response messages to JSON messages and to invoke (2) the outbound RESTful request and wait for the response.
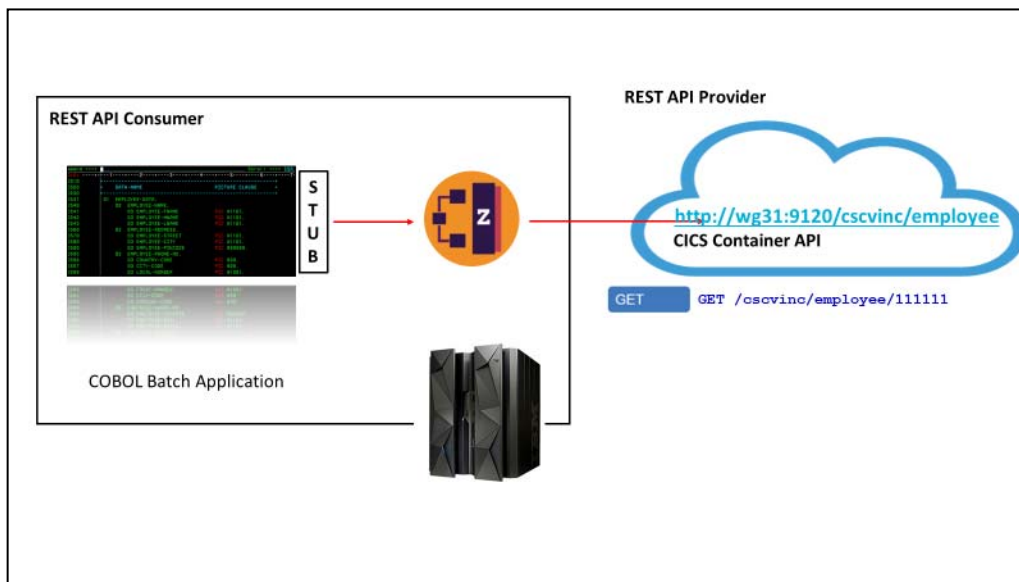
In this exercise, IMS TM programs will be compiled and executed in an IMS MPR environment. The only differences between CICS and MVS Batch and IMS is the configuration required for connecting to the z/OS Connect EE server. Consult the z/OS Connect EE Knowledge Center for information how this is done for CICS versus MVS Batch and IMS.

IMS TM RESTful API Requester to a VSAM API Provider

In this section an API requester will be developed and tested where the target API accesses a VSAM data set. The IMS MPP programs will invoke the API to insert (POST), retrieve (GET), update (PUT) and delete (DELETE) VSAM records.

Generate the API requester artifacts from a Swagger document

The first step is to use the API's Swagger document from the API to generate the artifacts required for the COBOL API client application and the API requester archive file. The Swagger document was obtained from the server where the API is hosted. All the files mention in this section are in directory *c:/z/apiRequester/ims* on your workstation.

The z/OS Connect EE build toolkit (ZCONBT) is a Java application shipped with z/OS Connect EE as a zip file and can be installed on a workstation or in OMVS. Anywhere a Java runtime is available. In this exercise the build toolkit will on Windows.

___1. Begin by reviewing the API's Swagger document for the VSAM API provider. This document identifies the types of HTTP methods supported as well as the path of each HTTP method. It also the contents each method's request JSON message and the response JSON message field contents and field types. This is the blueprint for invoking the API.

```
{
  "swagger" : "2.0",
  "info" : {
    "description" : "",
    "version" : "2.0.0",
    "title" : "filea"
  },
  "host" : "localhost:8080",
  "basePath" : "/employee",
  "schemes" : [ "https", "http" ],
  "consumes" : [ "application/json" ],
  "produces" : [ "application/json" ],
  "paths" : {
    "/file/{employee}" : {
      "get" : {
        "tags" : [ "filea" ],
        "operationId" : "getCscvincSelectService",
        "parameters" : [ {
          "name" : "Authorization",
          "in" : "header",
          "required" : false,
          "type" : "string"
        }, {
          "name" : "employee",
          "in" : "path",
          "required" : true,
          "type" : "string",
          "maxLength" : 6
```

Tech-Tip: Instructions for installing the z/OS Connect EE build toolkit can be found at URL
https://www.ibm.com/support/knowledgecenter/en/SS4SVW_3.0.0/com.ibm.zosconnect.doc/installing/bt_install.html

___2. Next review the input parameters that will be passed to the z/OS Connect EE build toolkit in a properties file. This file is *filea.properties* see its below:

```
apiDescriptionFile=./swagger.json  1
dataStructuresLocation=./syslib  2
apiInfoFileLocation=./syslib  3
logFileDirectory=./logs  4
language=COBOL  5
connectionRef=fileaAPI  6
requesterPrefix=ims  7
```

1. Identifies the Swagger document that describes the API

2. Provides the directory where the generated COBOL copybooks representing the request and response messages will be written.

3. Provides the directory where the generated COBOL copybook with invocation details of the API will be written.

4. Provides the directory where a log file will be written.

5. Identifies the target language.

6. Identifies the corresponding *zosconnect_endpointConnection* element define in the z/OS Connect EE server's server.xml file.

7. Provides a 3-character prefix to be used for all generated copybooks

___3. View file *filea.bat* which contains this one line.

```
c:\z\zconbt\bin\zconbt.bat -p=./filea.properties -f=./filea.ara
```

This is batch command file that when executed invokes the z/OS Connect EE build toolkit *(zconbt)*. The -p switch identifies the input properties file and the -f switch identifies the name of the generate API requester archive file.   The API requester archive (ARA) file will be deployed and made available to the z/OS Connect EE server in directory *…/resources/zosconnect/apiRequsters.*

___4. On the workstation desktop, locate the *Command Prompt* icon and double click on it to open a DOS session that will allow commands to be entered.

___5. Use the change directory command (cd) to change to directory C:\z*api\Requester\ims* e.g.
  cd c:\z\apiRequester\ims

___6. Enter command *filea* in the DOS command prompt window and you should see output like the below:

```
BAQB0000I: z/OS Connect Enterprise Edition Build Toolkit Version 1.2 (20200408-1120).
BAQB0008I: Creating API requester archive from configuration file ./filea.properties.
BAQB0040I: The generated API requester is automatically named filea_2.0.0 based on the title filea
and version 2.0.0 of the API to be called.

DFHPI9586W A reserved word "address" has been detected in the input document, it has been changed
to "Xaddress".
DFHPI9586W A reserved word "date" has been detected in the input document, it has been changed to
"Xdate".
BAQB0016I: Successfully processed operation (operationId: getCscvincSelectService, relativePath:
/file/{employee}, method: GET).
BAQB0015I: Start processing operation (operationId: putCscvincUpdateService, relativePath:
/file/{employee}, method: PUT).
DFHPI9586W A reserved word "address" has been detected in the input document, it has been changed
to "Xaddress".
DFHPI9586W A reserved word "date" has been detected in the input document, it has been changed to
"Xdate".
BAQB0016I: Successfully processed operation (operationId: putCscvincUpdateService, relativePath:
/file/{employee}, method: PUT).
BAQB0015I: Start processing operation (operationId: deleteCscvincDeleteService, relativePath:
/file/{employee}, method: DELETE).
BAQB0016I: Successfully processed operation (operationId: deleteCscvincDeleteService, relativePath:
/file/{employee}, method: DELETE).
BAQB0015I: Start processing operation (operationId: postCscvincInsertService, relativePath:
/insert, method: POST).
.
DFHPI9586W A reserved word "address" has been detected in the input document, it has been changed
to "Xaddress".
DFHPI9586W A reserved word "date" has been detected in the input document, it has been changed to
"Xdate".
BAQB0016I: Successfully processed operation (operationId: postCscvincInsertService, relativePath:
/insert, method: POST).

Total 4 operation(s) (success: 4, ignored: 0) defined in api description file: ./swagger.json
------ Successfully processed operation(s) ------
operationId: getCscvincSelectService, basePath: /employee, relativePath: /file/{employee}, method:
GET
- request data structure  : IMS00Q01
- response data structure : IMS00P01
- api info file           : IMS00I01

operationId: putCscvincUpdateService, basePath: /employee, relativePath: /file/{employee}, method:
PUT
- request data structure  : IMS01Q01
- response data structure : IMS01P01
- api info file           : IMS01I01

operationId: deleteCscvincDeleteService, basePath: /employee, relativePath: /file/{employee},
method: DELETE
- request data structure  : IMS02Q01
- response data structure : IMS02P01
- api info file           : IMS02I01

operationId: postCscvincInsertService, basePath: /employee, relativePath: /insert, method: POST
- request data structure  : IMS03Q01
- response data structure : IMS03P01
- api info file           : IMS03I01


BAQB0009I: Successfully created API requester archive file ./filea.ara.
```

Note that each HTTP method defined in the Swagger document will cause the generation of up to three copy books. One for the request message(Q01), one for the response message(P01) and one for the API details(I01). The copy book names are based on the *requesterPrefix* property (e.g. *csc*) and use an ascending sequence number sequence to differentiate between methods. Also note that there may be fewer than three copy books generated. For example, if there is no response message or no request message for a specific method a copy book may not be generated.

> **Important Note:** The COBOL programs used in this exercise include these generated copy books with the names as generated above, e.g. the GET COBOL program includes CSC01 prefixed copy books. Occasionally (probably after applying service) the sequence of the methods will change causing the corresponding sequence number of the copy book to also change. Review the sequence of the processing of the methods and change the COBOL applications so they include the correct copy book. Otherwise the compilation of the COBOL program may fail with errors or unexpected results may be observed.

___7. Now explore a sample snippet of the copy book for the request message of a PUT method. Use *EditPad Lite* to open file c:\z\apiRequester\ims\syslib\IMS01Q01. Scroll down until you see something like the code below.

```
        06 ReqPathParameters.
          09 employee-length                     PIC S9999 COMP-5 SYNC.
          09 employee                            PIC X(6).
        06 ReqBody.
           09 cscvincUpdateServiceOp-num         PIC S9(9) COMP-5 SYNC.
           09 cscvincUpdateServiceOperatio.
            12 cscvincContainer.
              15 request2-num                     PIC S9(9) COMP-5    SYNC.
              15 request.
                18 filea2-num                     PIC S9(9) COMP-5 SYNC.
                18 filea.
                  21 name-num                     PIC S9(9) COMP-5 SYNC.
                  21 name.
                   24 name2-length                PIC S9999 COMP-5 SYNC.
                   24 name2                        PIC X(20).
                  21 Xaddress-num                 PIC S9(9) COMP-5 SYNC.
                  21 Xaddress.
                   24 Xaddress2-length            PIC S9999 COMP-5 SYNC.
                   24 Xaddress2                    PIC X(20).
                  21 phoneNumber-num              PIC S9(9)  COMP-5 SYNC.
                  21 phoneNumber.
                   24 phoneNumber2-length         PIC S9999 COMP-5 SYNC.
                   24 phoneNumber2                 PIC X(8).
                  21 Xdate-num                    PIC S9(9) COMP-5 SYNC.
                  21 Xdate.
                   24 Xdate2-length               PIC S9999 COMP-5 SYNC.
                   24 Xdate2                        PIC X(8).
                  21 amount-num                   PIC S9(9) COMP-5 SYNC.
                  21 amount.
                   24 amount2-length              PIC S9999 COMP-5 SYNC.
                   24 amount2                       PIC X(8)
```

Mitch Johnson (mitchj@us.ibm.com)

Note that each variable has an associated length (e.g. *name2-length).* The length of a variable is required since there is no delimiter between the variables in storage. The runtime needs to know the size of a variable at execution time so the request and response messages can be properly converted to and from JSON name/value pairs.

For this API, the Swagger document included an additional variable for the number of occurrences of a variable, (e.g. *name-num*). In this exercise we will set the number of each variable to one.

> **Tech-Tip:** This additional variable was included for this API because the target program is a CICS container enabled program and a CICS channel can have multiple occurrences of the same container. The z/OS Connect EE API requester support needs to know number of each variable is being sent (and returned). This behavior is not unique to CICS.

The corresponding response message (*c:\z\apiRequester\ims\syslib\IMS01P01*) will have a similar layout.

___8. The corresponding API information copy book (*c:\z\apiRequester\ims\\syslib\IMS02I01*) has contents providing the API name and the path to be used for the method:

```
03 BAQ-APINAME              PIC X(255)
   VALUE 'filea_2.0.0'.
03 BAQ-APINAME-LEN          PIC S9(9) COMP-5 SYNC
   VALUE 11.
03 BAQ-APIPATH              PIC X(255)
   VALUE '%2Femployee%2Ffile%2F%7Bemployee%7D'.
03 BAQ-APIPATH-LEN          PIC S9(9) COMP-5 SYNC
   VALUE 35.
03 BAQ-APIMETHOD            PIC X(255)
   VALUE 'PUT'.
03 BAQ-APIMETHOD-LEN        PIC S9(9) COMP-5 SYNC
   VALUE 3.
```

Deploy the API Requester Archive file to the z/OS Connect EE server

Next make the API requester archive file available to the z/OS Connect EE server.

The API requester archive (ARA) file needs to be deployed to the API requester directory. In this case the target directory is */var/ats/zosconnect/servers/zceeapir/resources/zosconnect/apiRequesters*.

___1. Open a DOS command session and change to directory c:\z\apiRequester\ims, e.g **cd c:\z\apiRequester\ims**

___2. Use the z/OS Connect EE RESTful administrative interface to deploy the ARA files by using the cURL command embedded in command file *deploy.cmd*. Invoke this command file to deploy the filea API archive file.

```
c:\z\apiRequester\ims>curl -X POST --user Fred:fredpwd --data-binary @filea
c.ara  --header "Content-Type: application/zip" --insecure https://wg31.washingt
on.ibm.com:9483/zosConnect/apiRequesters
{"name":"filea_2.0.0","version":"1.0.0","description":"","status":"Started","apiRe
questerUrl":"https:\/\/wg31.washington.ibm.com:9483\/zosConnect\/apiRequeste
rs\/filea_2.0.0","connection":"fileaAPI"}
```

Mitch Johnson (mitchj@us.ibm.com)

> **Tech-Tip:** If a REST client tool like cURL or Postman was not available then ARA file could have been deployed using FTP to upload the file in binary mode to the *apiRequesters* directory.

> **Tech-Tip:** If an ARA needs to be redeployed, the cURL command with a PUT method can be used to stop the API requester
> *curl -X PUT --user Fred:fredpwd --insecure*
> *https://wg31.washington.ibm.com:9483/zosConnect/apiRequesters/filea_2.0.0?status=stopped*
> And the cURL command with a DELETE method can be used to delete the API requester.
> *curl -X DELETE --user Fred:fredpwd --insecure*
> *https://wg31.washington.ibm.com:9483/zosConnect/apiRequesters/filea_2.0.0*
> otherwise the redeployment of the ARA file will fail.

___3. The z/OS Connect EE server has the configuration below already included.

```
<server description="API Requester">
   <!-- Enable features -->
   <featureManager>
       <feature>zosconnect:apiRequester-1.0</feature>
   </featureManager>

 <zosconnect_endpointConnection id="fileaAPI"        1
       host="http://wg31.washington.ibm.com"
       port="9120"
       basicAuthRef="myBasicAuth"
       connectionTimeout="10s"
       receiveTimeout="20s" />

 <zosconnect_authData id="myBasicAuth"        2
       user="Fred"
       password="fredpwd" />

</server>
```

1. Identifies the system which hosts the target API as well as any security information and time out parameters. Note that the ID of this element matches the value *connectionRef* property used when the ARA was generated by the z/OS Connect build toolkit.

2. The target server uses basic authentication, so we are simply provided a user identity and password. The password can be encrypted in the server.xml file or TLS used for security.

> **Tech-Tip:** The *endpointConnection* element specifies the host and port of the target API provider. This z/OS Connect server performs the conversion of the COBOL data structure to JSON and then acts as a JSON REST client to the target API provider. Which for this example, is the same z/OS Connect server. The REST client loops back to itself when invoking the API. This would rarely happen in a real-world environment. This is why this host and port is also used by the API requester client in its environment variables.

Move the COBOL copy books to an MVS data set

Next make the generated COBOL copy books available for including into the COBOL program.

The generated COBOL copy books need to be available in an MVS data set which is included in the compiler SYSLIB concatenation sequence. In the data set is *USER1.ZCEE. SOURCE*. Use a DOS command prompt session and the commands below to move the copy books file to this data set. Authenticate as user USER1 (password USER1).

___1. Open a DOS command prompt and use the change directory command to go to directory C:\z\apiRequester\ims\syslib, e.g. **cd \z\apiRequester\ims\syslib**

___2. Start a file transfer session with the WG31 host using the *ftp* command, e.g. **ftp wg31**

___3. Logon as USER1 and then use the *cd* command to change to directory to data set *USER1.ZCEE. SOURCE, e.g.* **cd zcee.source**

___4. Toggle prompting off by entering command **prompt**

___5. Perform multiple put requests by using the multiple put command, **mput ims***

___6. When the last transfer has completed enter the **quit** command.

```
c:\z\apiRequester\ims>cd syslib
c:\z\apiRequester\ims\syslib>ftp wg31.washington.ibm.com
Connected to wg31.washington.ibm.com.
220-FTP 16:26:23 on 2018-02-15.
220 Connection will close if idle for more than 200 minutes.
User (wg31.washington.ibm.com:(none)): user1
331 Send password please. user1
Password:
230 USER1 is logged on.  Working directory is "USER1.".
ftp> cd zcee.source
250 The working directory "USER1.ZCEE.SOURCE" is a partitioned data set
ftp> prompt
Interactive mode Off .
ftp> mput IMS*
200 Port request OK.
125 Storing data set USER1.ZCEE.SOURCE(IMS00I01)
250 Transfer completed successfully.
ftp: 533 bytes sent in 0.32Seconds 1.66Kbytes/sec.
200 Port request OK.
sent in 0.10Seconds 110.89Kbytes/sec.
200 Port request OK.
…….
125 Storing data set USER1.ZCEE.SOURCE(IMS03P01)
250 Transfer completed successfully.
ftp: 7393 bytes sent in 0.32Seconds 22.89Kbytes/sec.
200 Port request OK.
125 Storing data set USER1.ZCEE.SOURCE(IMS03Q01)
250 Transfer completed successfully.
ftp: 1132 bytes sent in 0.40Seconds 2.80Kbytes/sec.
ftp> quit
```

Review the application programs

Now that the copy books have been generated and are available in a data set the next step is compile and link edit the application programs. The source for the application programs is in USER1.ZCEE.SOURCE. Member CSCVDLTI invokes the DELETE method, member CSCVGETI invokes the GET method, CSCVPSTI invokes the POST method and CSCVPUTI invokes the PUT method.

> **Tech-Tip:** Data set USER1.ZCEE.SOURCE can be accessed either by using the IBM z/OS Explorer or by using the Personal Communication icon on the desktop and logging on to TSO and use ISPF option 3.4 to access this data set. If you have any questions about either method, ask the instructor.

___1. Back in the IBM z/OS Explorer session switch to the *Remote System Explorer* perspective and select *wg31.washington.ibm.com* under *z/OS Remote System* and use the **Connect** button to connect to the remote system explorer daemon running on z/OS.
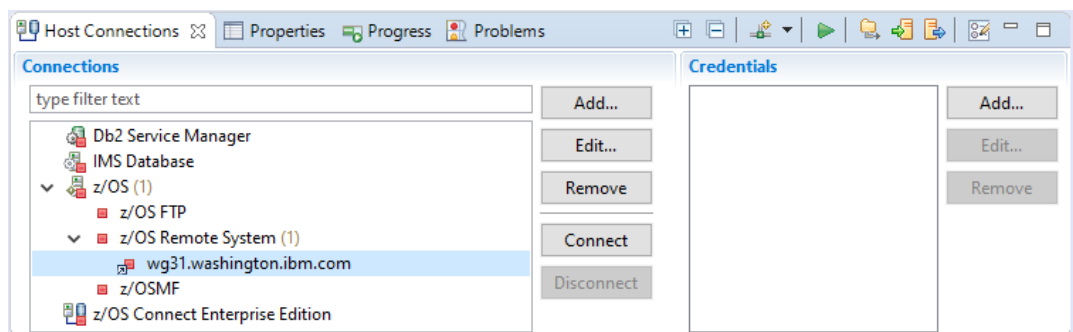
___2. Expand the USER1.ZCEE filter and select data set *USER1.ZCEE.SOURCE* in the *Remote System view* and double click to display a list of its members.
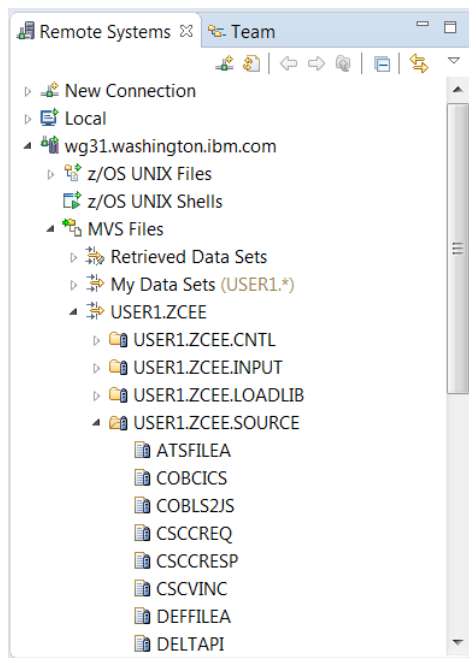
Tech-Tip: Data set USER1.ZCEE.SOURCE can be accessed either by using the IBM z/OS Explorer or by using the Personal Communication icon on the desktop and logging on to TSO and use ISPF option 3.4 to access this data set. If you have any questions about either method, ask the instructor.

___3. Begin by reviewing the application programs CSCVGETI. Open member *CSCVGETI* in *USER1.ZCEE. SOURCE* and review its contents by selecting the member and right- mouse button clicking and selecting the *Open* option. Scroll down until you see these lines of code.

These lines of code copy into the source the required z/OS Connect EE copy book (BAQRINFO) and the copy books generated by the z/OS Connect EE build tool kit. If when you generated the copy book earlier in the exercise and the sequence number used for the GET method was not 00, e.g. the IMSC**00** prefix, then you need to change the COBOL source to match sequence generated when you invoke the ZCONBT tool.

```
       * Copy API Requester required copybook
        COPY BAQRINFO.

       * Request and Response
        01 GET-REQUEST.
           COPY IMS01Q01.
        01 GET-RESPONSE.
           COPY IMS01P01.
       * Structure with the API information
        01 GET-INFO-OPER1.
           COPY IMS01I01.
```

___4. Scroll down further until you find this code which provides the contents of the request message.

```
       *----------------------------------------------------------------*
       * Set up the data for the API Requester call                     *
       *------------------- --------------------------------------------*
           MOVE employee of PARM-DATA TO employee IN GET-REQUEST.
           MOVE LENGTH of employee in GET-REQUEST to
               employee-length IN GET-REQUEST.

       *----------------------------------------------------------------*
       * Initialize API Requester PTRs & LENs                           *
       *----------------------------------------------------------------*
       * Use pointer and length to specify the location of
       *  request and response segment.
       * This procedure is general and necessary.
           SET BAQ-REQUEST-PTR TO ADDRESS OF GET-REQUEST.
           MOVE LENGTH OF GET-REQUEST TO BAQ-REQUEST-LEN.
           SET BAQ-RESPONSE-PTR TO ADDRESS OF GET-RESPONSE.
           MOVE LENGTH OF PUT-RESPONSE TO BAQ-RESPONSE-LEN.
```

The *GET* method only requires one variable and it is provided as a path parameter, e.g. /employee/{numb}.

___5. Scroll down further and you will the call to the z/OS Connect API requester stub module.

```
         *----------------------------------------------------------------*
         * Call the communication stub                                    *
         *----------------------------------------------------------------*
         * Call the subsystem-supplied stub code to send
         * API request to zCEE
             CALL COMM-STUB-PGM-NAME USING
                     BY REFERENCE    GET-INFO-OPER1
                     BY REFERENCE    BAQ-REQUEST-INFO
                     BY REFERENCE    BAQ-REQUEST-PTR
                     BY REFERENCE    BAQ-REQUEST-LEN
                     BY REFERENCE    BAQ-RESPONSE-INFO
                     BY REFERENCE    BAQ-RESPONSE-PTR
                     BY REFERENCE    BAQ-RESPONSE-LEN.
         * The BAQ-RETURN-CODE field in 'BAQRINFO' indicates whether this
         * API call is successful.
```

Note that the parameters are being passed by reference. This means that the z/OS Connect API requester stub will have direct access to the program storage for both accessing data and making changes (i.e. the response message).

Mitch Johnson (mitchj@us.ibm.com)

___6. Scroll down further and you will see the code where the application displays the results in the response message.

```
        IF BAQ-SUCCESS THEN
            MOVE name2         of GET-RESPONSE TO NAME  of OUT-BUFFER
            MOVE Xaddress2     of GET-RESPONSE TO ADDRX  of OUT-BUFFER
            MOVE phoneNumber2 of GET-RESPONSE TO PHONE  of OUT-BUFFER
            MOVE Xdate2        of GET-RESPONSE TO DATEX  of OUT-BUFFER
            MOVE amount2       of GET-RESPONSE TO AMOUNT of OUT-BUFFER
            MOVE USERID2       of GET-RESPONSE TO USERID of OUT-BUFFER
```

Close member *CSCVGETI* and open member *CSCVPSTI*. The code for this program is very similar to the code for *GETAPI* with the exception that all variables are provided in a JSON request message. Scroll down and find the code below.

```
        *----------------------------------------------------------
        * Set up the data for the API Requester call
        *----------------------------------------------------------
            MOVE NUMB of IN-BUFFER to NUMB of OUT-BUFFER.
            MOVE 1 to cscvincInsertServiceOp-num of ReqBody,
                request2-num, filea2-num, employeeNumber-num, name-num,
                Xaddress-num, phoneNumber-num, Xdate-num, amount-num.
            MOVE NUMB of IN-BUFFER TO employeeNumber2 IN POST-REQUEST.
            MOVE LENGTH of employeeNumber2 in POST-REQUEST to
                employeeNumber2-length IN POST-REQUEST.
            MOVE NAME of IN-BUFFER TO name2 IN POST-REQUEST.
            MOVE LENGTH of name2 in POST-REQUEST to
                name2-length IN POST-REQUEST.
            MOVE ADDRX of IN-BUFFER TO Xaddress2 IN POST-REQUEST.
            MOVE LENGTH of Xaddress2 in POST-REQUEST to
                Xaddress2-length IN POST-REQUEST.
```

In this case the number of occurrences of a property must be provided and the property value moved to the request copy book.

___7. Explore members *CSCVDLTI* and *CSCVPUTI* and their corresponding copy books to see the code which is common (variable housekeeping, calls to the z/OS Connect stub, etc.) regardless of the method being invoked.

Mitch Johnson (mitchj@us.ibm.com)

Compile and link-edit the application programs

The applications programs now are ready to be compiled and link edited.

___1. Submit the job in member **CSCVIMS** in data set *USER1.ZCEE.CNTL.*

```
//USER1S JOB (ACCOUNT),USER1,NOTIFY=&SYSUID,REGION=0M,
// CLASS=A,MSGCLASS=H,MSGLEVEL=(1,1)
//COMPILE EXEC IGYWCL,LNGPRFX=SYS1.ECOBOL,PARM.COBOL='NODYNAM'
//COBOL.SYSIN  DD DISP=SHR,DSN=USER1.ZCEE.SOURCE(CSCVGETI)
//COBOL.SYSLIB DD DISP=SHR,DSN=USER1.ZCEE.SOURCE
//             DD DISP=SHR,DSN=ZCEE30.SBAQCOB
//LKED.SYSLMOD DD DISP=SHR,DSN=JOHNSON.ZCEE.SDFHLOAD(CSCVGETI)
//RESLIB   DD DSN=DFSF10.SDFSRESL,DISP=SHR
//SYSIN    DD *
 INCLUDE RESLIB(DFSLI000)
 NAME  CSCVGETI(R)
//COMPILE EXEC IGYWCL,LNGPRFX=SYS1.ECOBOL,PARM.COBOL='NODYNAM'
//COBOL.SYSIN  DD DISP=SHR,DSN=USER1.ZCEE.SOURCE(CSCVDLTI)
//COBOL.SYSLIB DD DISP=SHR,DSN=USER1.ZCEE.SOURCE
//             DD DISP=SHR,DSN=ZCEE30.SBAQCOB
//LKED.SYSLMOD DD DISP=SHR,DSN=JOHNSON.ZCEE.SDFHLOAD(CSCVDLTI)
//RESLIB   DD DSN=DFSF10.SDFSRESL,DISP=SHR
//SYSIN    DD *
 INCLUDE RESLIB(DFSLI000)
 NAME  CSCVDLTI(R)
//COMPILE EXEC IGYWCL,LNGPRFX=SYS1.ECOBOL,PARM.COBOL='NODYNAM'
//COBOL.SYSIN  DD DISP=SHR,DSN=USER1.ZCEE.SOURCE(CSCVPSTI)
//COBOL.SYSLIB DD DISP=SHR,DSN=USER1.ZCEE.SOURCE
//             DD DISP=SHR,DSN=ZCEE30.SBAQCOB
//LKED.SYSLMOD DD DISP=SHR,DSN=JOHNSON.ZCEE.SDFHLOAD(CSCVPSTI)
//RESLIB   DD DSN=DFSF10.SDFSRESL,DISP=SHR
//SYSIN    DD *
 INCLUDE RESLIB(DFSLI000)
 NAME  CSCVPSTI(R)
//COMPILE EXEC IGYWCL,LNGPRFX=SYS1.ECOBOL,PARM.COBOL='NODYNAM'
//COBOL.SYSIN  DD DISP=SHR,DSN=USER1.ZCEE.SOURCE(CSCVPUTI)
//COBOL.SYSLIB DD DISP=SHR,DSN=USER1.ZCEE.SOURCE
//             DD DISP=SHR,DSN=ZCEE30.SBAQCOB
//LKED.SYSLMOD DD DISP=SHR,DSN=JOHNSON.ZCEE.SDFHLOAD(CSCVPUTI)
//RESLIB   DD DSN=DFSF10.SDFSRESL,DISP=SHR
//SYSIN    DD *
 INCLUDE RESLIB(DFSLI000)
 NAME  CSCVPUTI(R)
```

Note that the CALL statement in each of this program uses a variable for the program name z/OS Connect EE stub program (BAQCSTUB). This means that this program will by dynamically linked at execution. The load module library containing BAQSCTUB (ZCEE30.SBAQLIB) must be available in the STEPLIB concatenation sequence when the program is executed.
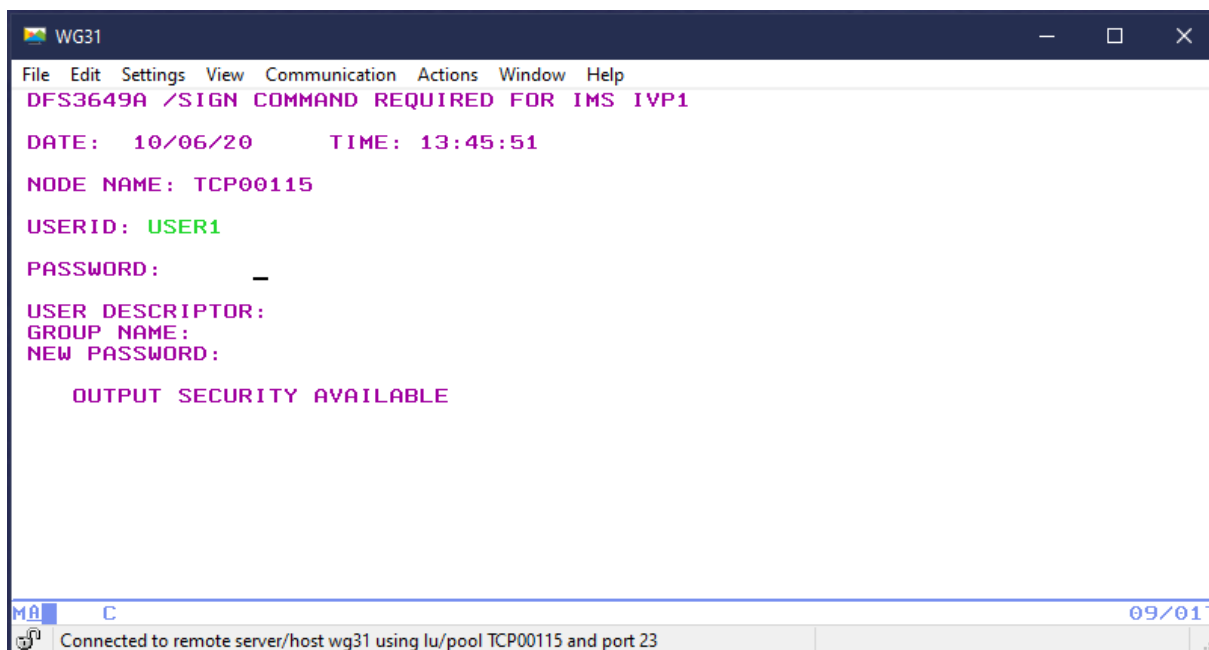
This job should complete with a zero-return code.

Mitch Johnson (mitchj@us.ibm.com)

> **Tech Tip:** To submit a job when using the IBM z/OS Explorer select the member and right mouse button click and select the *Submit* option. Or if the member is currently opened simply right mouse button click and select the *Submit* option. Click the **Locate Job** button on the *Job Confirmation* pop-up. This will display the job output in the *Retrieved Job* section under *JES* in the *Remote Systems* view. The job's output can be viewed right mouse button clicking and selecting the *Open* option.

Test the API requester application programs

___1. In a new or existing 3270 session start a session with IMS by entering *IMS* on a new 3270 PCCOM session.

___2. On sign on screen, enter *USER1* as the *USERID* and USER1's password and press **Enter** to continue.

```
WG31                                                     —   □   ×

File  Edit  Settings  View  Communication  Actions  Window  Help
DFS3649A /SIGN COMMAND REQUIRED FOR IMS IVP1

DATE:   10/06/20      TIME:  13:45:51

NODE NAME: TCP00115

USERID: USER1

PASSWORD:            _

USER DESCRIPTOR:
GROUP NAME:
NEW PASSWORD:

    OUTPUT SECURITY AVAILABLE




MA    C                                                09/017
   Connected to remote server/host wg31 using lu/pool TCP00115 and port 23
```

> **Tech Tip:** The IMS message processing region used for the CSCVGET, CSCVPUT, CSCVPST and CSCVCDLT transactions was modified by adding the z/OS Connect target data set SBAQLIB to the STEPLIB concatenation sequence and a new DD statement for CEEOPTS was added, as shown below:
>
> ```
> //CEEOPTS  DD *
>  POSIX(ON),
>  ENVAR("BAQURI=wg31.washington.ibm.com",
>  "BAQPORT=9120")
> ```

Mitch Johnson (mitchj@us.ibm.com)

___4. Clear the screen again and enter IMS transaction */FOR CSCVGET* to display the *EMPLOYEE INQUIRE DEMOSTATION* MFS screen shown below.
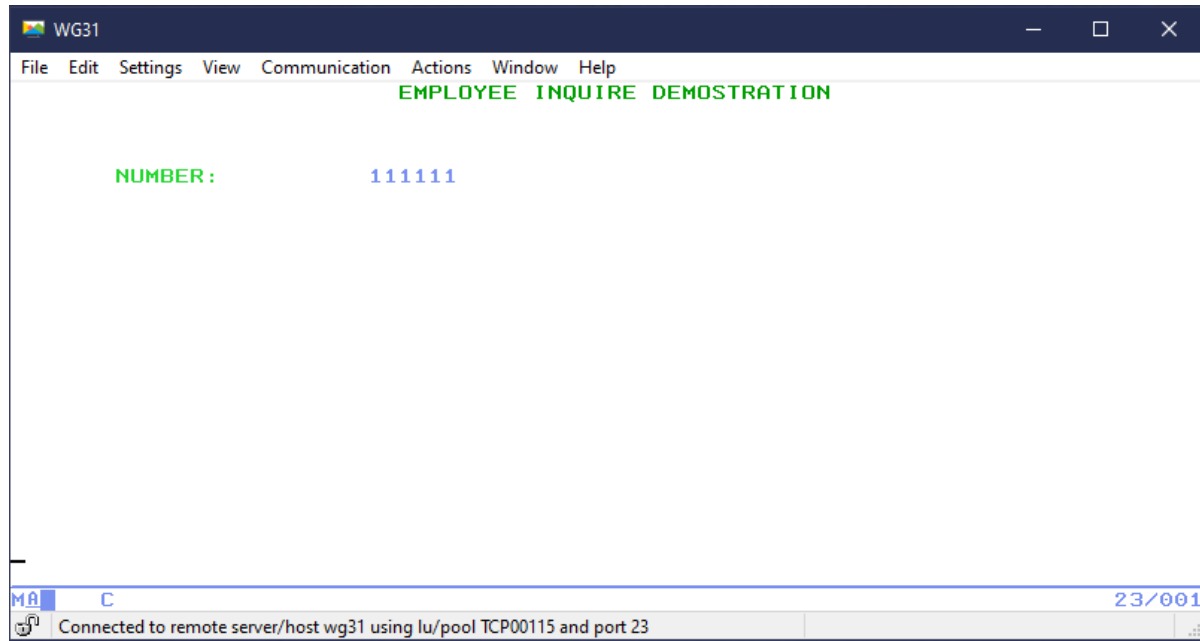
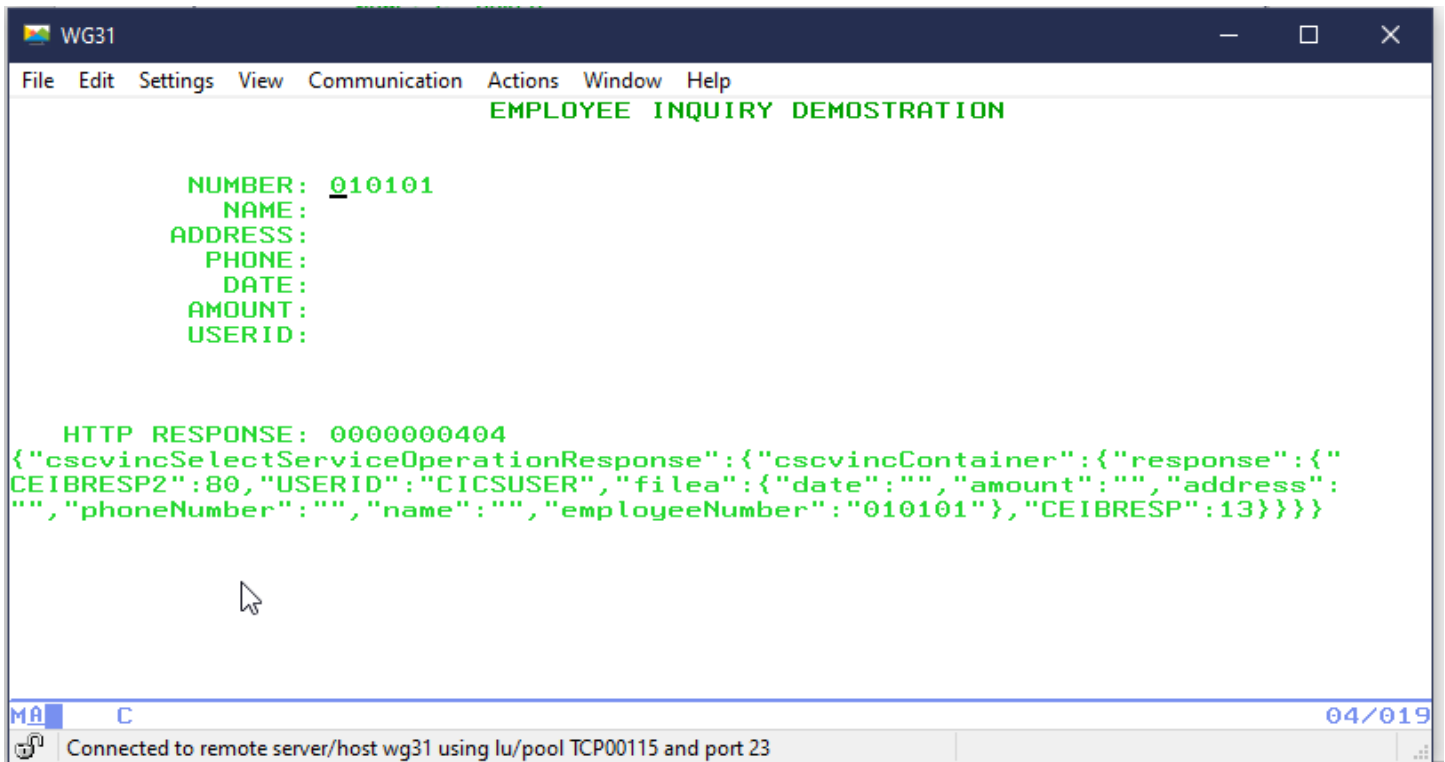___5. Enter *111111* as the value for *NUMBER* and press **ENTER**. This should invoke the *CSCVGET* program and invoke the GET method of the target API. The results will be returned and display in the MFS screen as shown below.
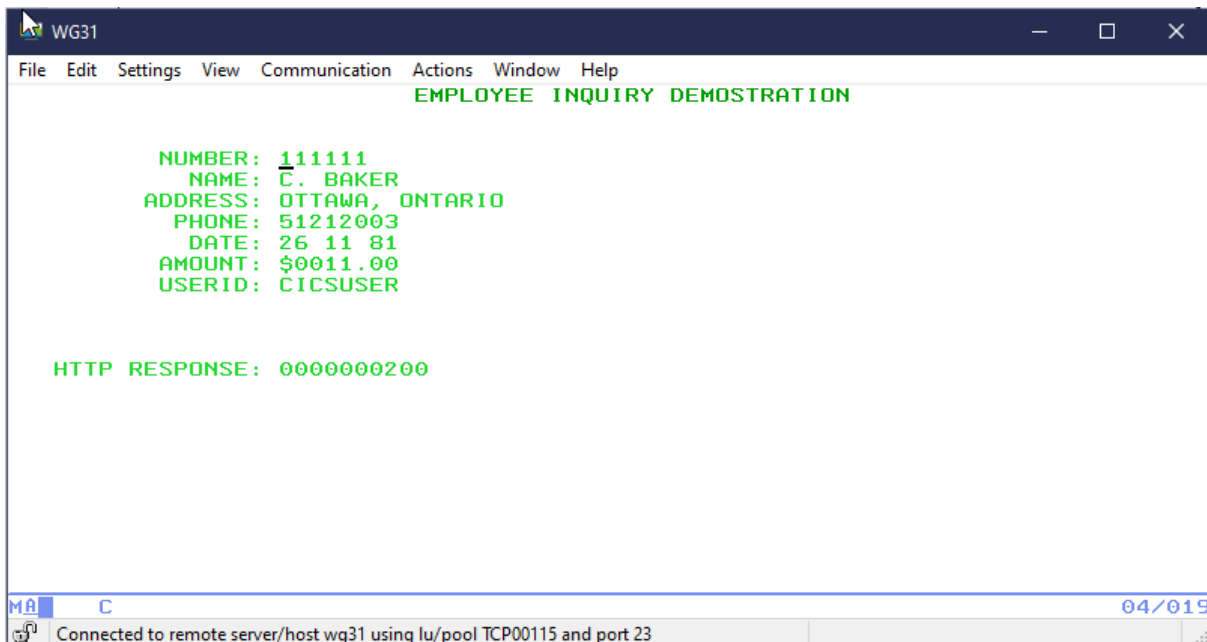
Tech-Tip: An HTTP code of 200 indicates success. For an explanation of HTTP codes see URL
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes

___6. Clear the screen again and enter IMS transaction */FOR CSCVGET* to display the *EMPLOYEE INQUIRE DEMOSTATION* MFS screen again but this time enter a value of ***010101*** for *NUMBER*. Pressing **Enter** should return the results below. The API sets the HTTP response code to *404* if the requested record does not exist and the program *CSCVGETI* display the error response message on the MFS screen.

___7. Optional. For debugging purposes, the IMS programs display information as they execute. Use SDSF to access the IMSMSG task and look for the SYSOUT output. You should see something like the below.

20201007100220 IOPCB ModNAME: CSCVGET
20201007100220 IOPCB Userid: USER1
20201007100220 IN-LL: 020
20201007100220 IN-BUFFER: CSCVGETI 111111
20201007100220 HTTP CODE: 0000000200
20201007100241 IOPCB ModNAME: CSCVGET
20201007100241 IOPCB Userid: USER1
20201007100241 IN-LL: 020
20201007100241 IN-BUFFER: CSCVGETI 010101
20201007100241 Error code: 0000000404
20201007100241 Error msg: {"cscvincSelectServiceOperationResponse":{"cscvincContainer":
{"response":{"CEIBRESP2":80,"USERID":"CICSUSER","filea":{"date":"","amount":"","address":"","phon
eNumber":"","name":"","employeeNumber":"010101"},"CEIBRESP":13}}}}

20201007100241 Error origin: API
20201007100241 HTTP CODE: 0000000404

___8. Clear the screen again and enter IMS transaction */FOR CSCVPST* to display the *INSERT EMPLOYEE DEMOSTATION* MFS screen. Enter a value of **010101** for *NUMBER* and enter data for the other fields.

___9. Pressing **Enter** should return the results below. The HTTP response code of *200* indicates the record was successfully inserted (POST) into the VSAM file.

___10. Repeating steps 6 and 7 with the same number should return these results.

The HTTP response of 409 (Conflict) indicates the record already exists and can be inserted again.

___11. Clear the screen again and enter IMS transaction */FOR CSCVPUT* to display the *EMPLOYEE UPDATE DEMOSTATION* MFS screen. Enter a value of **010101** for *NUMBER* and enter your name for the NAME field and other data for the other fields.

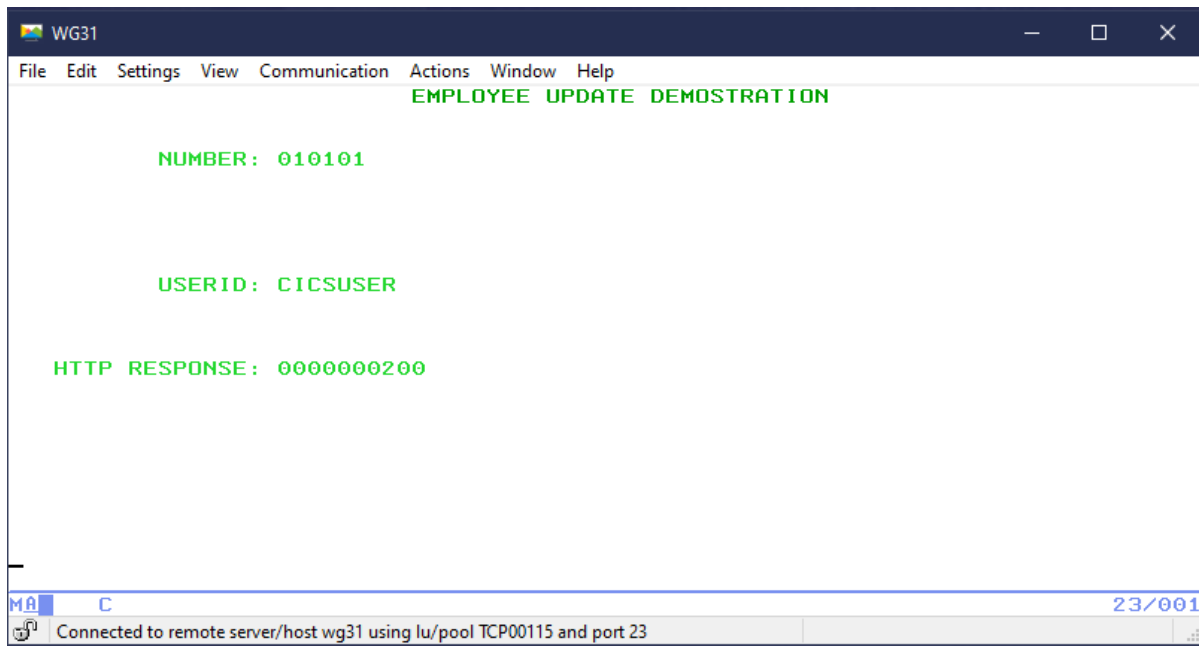Mitch Johnson (mitchj@us.ibm.com)

___12. Pressing **Enter** should return the results below. The HTTP response code of *200* indicates the record was successfully updated (PUT) into the VSAM file.

___13. Clear the screen again and enter IMS transaction */FOR CSCVGET* to display the *EMPLOYEE INQUIRE DEMOSTATION* MFS screen. Enter a value of *010101* for *NUMBER* and press **Enter**. You should see that the record now displays your name.

Mitch Johnson (mitchj@us.ibm.com)
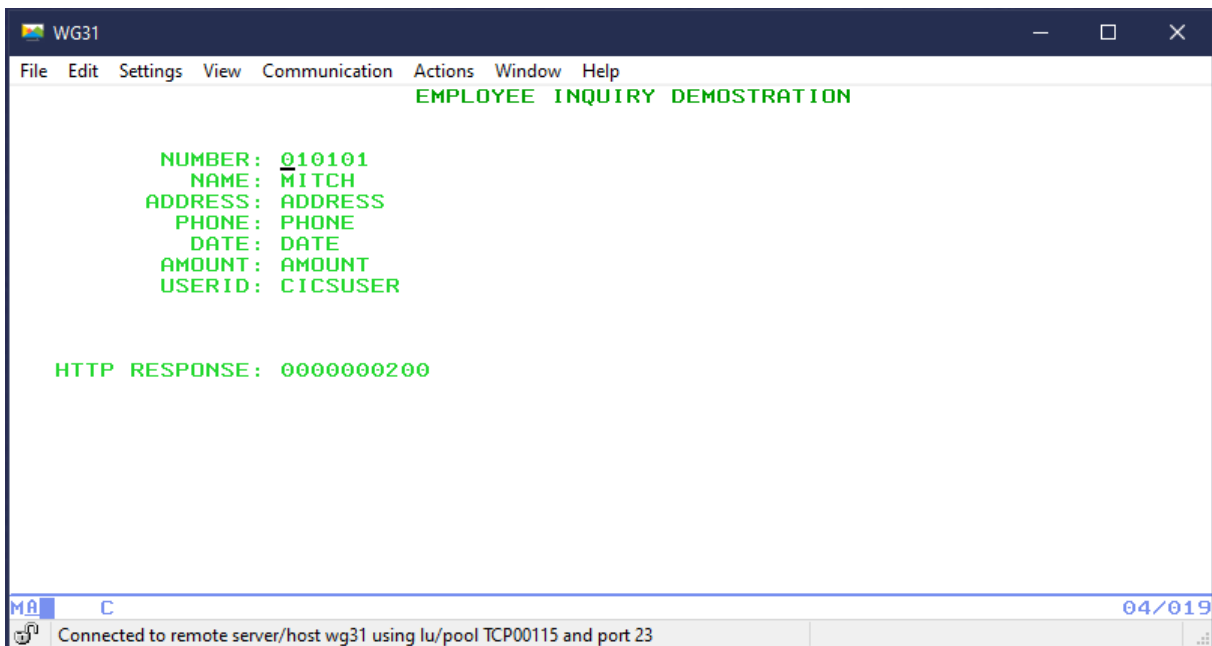
___14. Clear the screen again and enter IMS transaction */FOR CSCVDLT* to display the *DELETE EMPLOYEE DEMOSTATION* MFS screen. Enter a value of **010101** for *NUMBER* and press **Enter**. You should see a HTTP response code of 200 to indicate the record was successfully deleted.
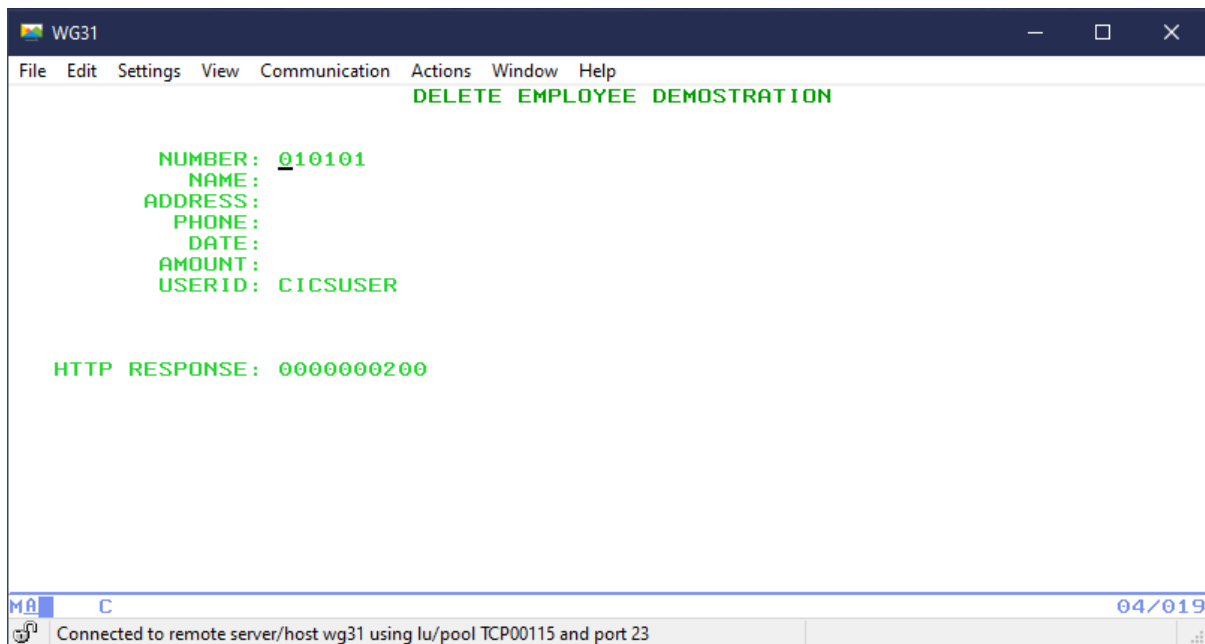
___15. Clear the screen again and again enter IMS transaction */FOR CSCVDLT* to display the *DELETE EMPLOYEE DEMOSTATION* MFS screen. Enter a value of **010101** for *NUMBER* and press **Enter**. You should see a HTTP response code of 404 to indicate the record was not found.

Mitch Johnson (mitchj@us.ibm.com)
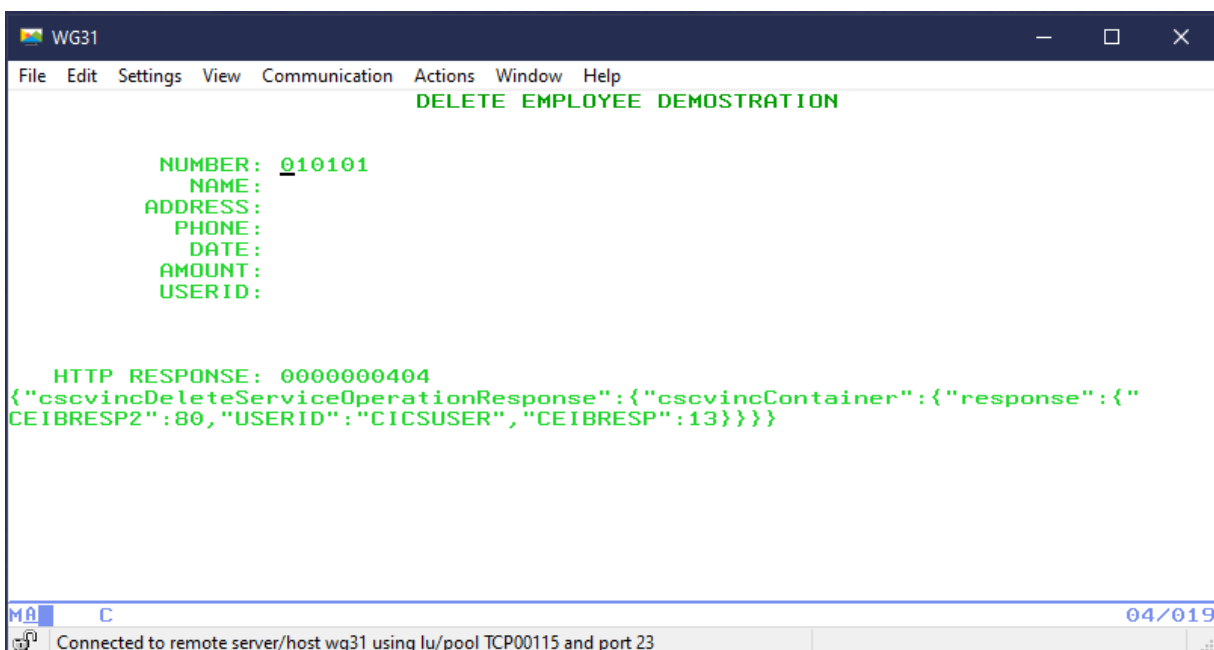
The available records are listed below:

numb	name	addrx	Phone	datex	amount
000100	S. D. BORMAN	SURREY, ENGLAND	32156778	26 11 81	$0100.11
000102	J. T. CZAYKOWSI	WARWICK, ENGLAND	98356183	26 11 81	$1111.11
000104	M. B. DOMBEY	LONDON,ENGLAND	12846293	26 11 81	$0999.99
000106	A. I. HICKSON	CROYDON, ENGLAND	19485673	26 11 81	$0087.71
000111	ALAN TULIP	SARATOGA,CALIFORNIA	46120753	01 02 74	$0111.11
000762	SUSAN MALAIKA	SAN JOSE,CALIFORNIA	22312121	01 06 74	$0000.00
000983	J. S. TILLING	WASHINGTON, DC	34512120	21 04 75	$9999.99
001222	D.J.VOWLES	BOBLINGEN, GERMANY	70315551	10 04 73	$3349.99
001781	TINA J YOUNG	SINDELFINGEN,GERMANY	70319990	21 06 77	$0009.99
003210	B.A. WALKER	NICE, FRANCE	12345670	26 11 81	$3349.99
003214	PHIL CONWAY	SUNNYVALE, CAL.	34112120	00 06 73	$0009.99
003890	BRIAN HARDER	NICE FRANCE	00000000	28 05 74	$0009.99
004004	JANET FOUCHE	DUBLIN, IRELAND	71112121	02 11 73	$1259.99
004445	DR. P. JOHNSON	SOUTH BEND, S.DAK.	61212120	26 11 81	$0009.99
004878	ADRIAN JONES	SUNNYVALE, CALIF.	32212120	10 06 73	$5399.99
005005	A. E. DALTON	SAN FRANCISCO, CA.	00000001	01 08 73	$0009.99
005444	ROS READER	SARATOGA, CALIF.	67712120	20 10 74	$0809.99
005581	PETE ROBBINS	BOSTON, MASS.	41312120	11 04 74	$0259.99
006016	SIR MICHAEL ROBERTS	NEW DELHI, INDIA	70331211	21 05 74	$0009.88
006670	IAN HALL	NEW YORK, N.Y.	21212120	31 01 75	$3509.88
006968	J.A.L. STAINFORTH	WARWICK, ENGLAND	56713821	26 11 81	$0009.88
007007	ANDREW WHARMBY	STUTTGART, GERMANY	70311000	10 10 75	$5009.88
007248	M. J. AYRES	REDWOOD CITY, CALF.	33312121	11 10 75	$0009.88
007779	MRS. A. STEWART	SAN JOSE, CALIF.	41512120	03 01 75	$0009.88
009000	P. E. HAVERCAN	WATERLOO, ONTARIO	09876543	21 01 75	$9000.00
100000	M. ADAMS	TORONTO, ONTARIO	03415121	26 11 81	$0010.00
111111	C. BAKER	OTTAWA, ONTARIO	51212003	26 11 81	$0011.00
200000	S. P. RUSSELL	GLASGOW, SCOTLAND	63738290	26 11 81	$0020.00
222222	DR E. GRIFFITHS	FRANKFURT, GERMANY	20034151	26 11 81	$0022.00
300000	V. J. HARRIS	NEW YORK, U.S.	64739801	26 11 81	$0030.00
333333	J.D. HENRY	CARDIFF, WALES	78493020	26 11 81	$0033.00
400000	C. HUNT	MILAN, ITALY	25363738	26 11 81	$0040.00
444444	D. JACOBS	CALGARY, ALBERTA	77889820	26 11 81	$0044.00
500000	P. KINGSLEY	MADRID, SPAIN	44454640	26 11 81	$0000.00
555555	S.J. LAZENBY	KINGSTON, N.Y.	39944420	26 11 81	$0005.00
600000	M.F. MASON	DUBLIN, IRELAND	12398780	26 11 81	$0010.00
666666	R. F. WALLER	LA HULPE, BRUSSELS	42983840	26 11 81	$0016.00
700000	M. BRANDON	DALLAS, TEXAS	57984320	26 11 81	$0002.00
777777	L.A. FARMER	WILLIAMSBURG, VIRG.	91876131	26 11 81	$0027.00
800000	P. LUPTON	WESTEND, LONDON	24233389	26 11 81	$0030.00
888888	P. MUNDY	NORTHAMPTON, ENG.	23691639	26 11 81	$0038.00
900000	D.S. RENSHAW	TAMPA, FLA.	35668120	26 11 81	$0040.00
999999	ANJI STEVENS	RALEIGH, N.Y.	84591639	26 11 81	$0049.00

When finished with your testing, disconnect from IMS by entering IMS transaction */RCL*

Summary

You have use the z/OS Connect build toolkit to generate an API requester archive file and the COBOL copy books that must be included in the COBOL application program when accessing the API requester archive file in a z/OS Connect server. The COBOL applications have been compiled and link-edited and the target API has been tested using various RESTful methods in an IMS/TM environment.