Downloads for TPF Family Products

Sample SOAP Consumer Application on z/TPF Enterprise Edition V1.1

Copyright International Business Machines Corporation, 2009. All Rights Reserved.

Note to US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Note: Before using this information and the product it supports, read the general information under "NOTICES" in this document.

CONTENTS

This file includes the following information:

1.0 ABOUT THIS README

This readme file will guide you through the process of downloading, installing, and using a sample SOAP consumer application and Web service stub on your z/TPF system. This sample application demonstrates how to use the z/TPF SOAP consumer application programming interfaces (APIs) (APAR PJ35511) to invoke a Web service on a remote platform or on the same z/TPF system.

The Sample SOAP Consumer package provides you with a complete sample that can be run on your z/TPF system. You can use it as a starting point for your own SOAP consumer applications and Web service stubs, use it for training purposes, or use it as-is. Go to the IBM TPF Product Information Center for more details about SOAP consumer support.

Note: The TPF development lab does not maintain this application and will not accept APARs on this code.

2.0 SYSTEM REQUIREMENTS

Before proceeding with these instructions, additional downloads may be required from the TPF Support Web site. Do the following:

1. Ensure that PJ35511 has been applied to your z/TPF system.
2. Download and install the Web Service Wrapper Sample application.
3. If you plan to use WebSphere MQ as the communications transport, download and install the sample WebSphere MQ communications binding for use by z/TPF provider Web service support.
4. If you plan to use HTTP as the communications transport, do the following:
   - Download and install the CURL open source library and enable z/TPF HTTP client support for use by the consumer Web service support.
   - If you are using the Apache 1.3 HTTP server, download and install the sample `mod_tpf_soap` communications binding for use with z/TPF provider Web service support.
   - If you are using the Apache 2.2 HTTP server, download and install the Apache 2.2 package, which includes a version of `mod_tpf_soap` for use with z/TPF provider Web service support.

3.0 DOWNLOADING

To download this module, do the following:

1. Click the **Download now** button to download the compressed sample SOAP consumer package (the tarball) to your PC. The name of this package is **soap_consumer_sample_zTPF.tar.Z**.

2. FTP the tarball to your home directory on your Linux system using binary mode:
   - Open an MS-DOS window and activate FTP by using the following command:
     ftp *your.linux.build.machine.com*
   - Sign in using your user name and password.
   - Set the mode to binary by entering the following command:
     binary
   - Send the file to your Linux system by using the following command:
     send c:*your_path*\soap_consumer_sample_zTPF.tar.Z soap_consumer_sample_zTPF.tar.Z
   - Exit FTP by entering the following command:
     bye

3. On your Linux system, create a working directory in your root directory by entering the following command:

   mkdir ~/your_workdir

4. Change to the working directory and extract the program files from the SOAP consumer sample package by entering the following command:

   cd ~/your_workdir
   tar -xzkf ../soap_consumer_sample_zTPF.tar.Z

   After you have completed this step, you will have the following files on your Linux system in the directory *~/your_workdir:*

   - Sample z/TPF Web service stub code:

        - qwc1.c
        - qwc1_fault.c
        - qwc1_request.c
        - qwc1_response.c
        - qwc1.h
    - MakeTPF sample Web service stub makefile (qwc1.mak)
    - Sample z/TPF SOAP consumer application code (qwc2.c)
    - MakeTPF sample application makefile (qwc2.mak)
    - Sample z/TPF SOAP consumer User transport code (qwc3_user_transport.c)
    - MakeTPF sample User transport makefile (qwc3.mak)
    - Sample z/TPF RESTful calculator Web service (qwc4_calc_service.c)
    - MakeTPF sample RESTful calculator Web service makefile (qwc4.mak)
5. Sample consumer Web service deployment descriptor for the sample application (CalculatorService_consumer.xml)
6. This readme (sampleconsumer_readme.htm).

4.0 COMPILING, LINKING, AND LOADING

1. Change to your working directory:

   cd ~/your_workdir

2. Create a `maketpf` configuration file named `maketpf.cfg`.
    - Ensure that the first assignment of `TPF_ROOT` in `maketpf.cfg` is the absolute path to your "*`~/your_workdir`*" directory.
    - Ensure that the first assignment of `APPL_ROOT` in `maketpf.cfg` is the absolute path to your "*`~/your_workdir`*" directory.
    - Update other fields (`TPF_BSS_NAME, TPF_SS_NAME, USER_VERSION_CODE`) if necessary.

3. Edit the sample `maketpf .mak` file for the sample Web service stub (qwc1.mak), for the sample SOAP consumer application (qwc2.mak), for the sample SOAP consumer User transport (qwc3.mak), and for the sample RESTful calculator Web service (qwc4.mak). Verify that the `maketpf_env` assignments in `qwc1.mak`, `qwc2.mak`, `qwc3.mak`, and `qwc1.mak` are correct for your build environment.

4. Compile and link the SOAP consumer sample programs.

   maketpf qwc1.mak -f
   maketpf qwc2.mak -f
   maketpf qwc3.mak -f
   maketpf qwc4.mak -f

5. Use the standard load procedure to transfer and load the SOAP consumer programs (QWC1, QWC2, QWC3, and QWC4) to your test system.

5.0 DEPLOYING

To deploy the sample Web service stub, making it accessible to the z/TPF SOAP handler, you will need to FTP the consumer Web service deployment descriptor to your z/TPF system and use the ZWSAT DEPLOY command. You will also need to define a mechanism for activating the sample SOAP consumer application (QWC2).

The sample is designed to run with both the SOAP consumer and SOAP provider on the same z/TPF processor. For example, the transport definitions in the consumer Web service deployment descriptor make use of the well-known loopback IP address (127.0.0.1) or rely on queue definitions that are local to the current z/TPF processor. If you wish to run the sample with the SOAP consumer and SOAP provider on separate z/TPF processors, update the consumer Web service deployment descriptor accordingly before continuing with these deployment steps.

1. FTP the consumer Web service deployment descriptor to the `/etc/tpf-ws/` directory on your z/TPF system using binary mode:
   - Change to the "*`~/your_workdir`*" directory:
     cd ~/your_workdir
   - FTP by using the following command:
     ftp *`your.zTPF.system`*
   - Sign in using your user name and password.
   - Set the mode to binary by entering the following command:
     binary
   - Send the file to your z/TPF system by using the following command:
     send CalculatorService_consumer.xml /etc/tpf-ws/CalculatorService_consumer.xml
   - Exit FTP by entering the following command:
     bye

2. On your z/TPF system, deploy the consumer Web service by entering the following command:

   ZWSAT DEPLOY DD-CalculatorService_consumer.xml

   Note: You must have completed 4.0 COMPILING, LINKING, AND LOADING before entering the ZWSAT DEPLOY command.

3. On your z/TPF system, define a temporary command that will be used to activate the sample SOAP consumer application by entering the following command:

   ZFMSG ADD ZCALC PROG-QWC2

4. Ensure that the communications binding(s) that you intend to use are correctly configured. For example:
   - If you are using the TPF-to-TPF transport, define and start the Internet daemon (INETD) listener for the TPF-to-TPF communications binding by entering the following commands:

     ZINET ADD S-ZTPFSOAP PGM-CCWS MODEL-AOR AORL-16 P-TCP PORT-1000 IP-ANY ACT-AUTO STATE-NORM

     ZINET START S-ZTPFSOAP

- If you are using the HTTP transport, define and start the INETD listener for the Apache HTTP server. Ensure the Apache configuration includes "mod_tpf_soap". Please refer to the Apache readme for more information.

- If you are using the WebSphere MQ transport, define and start the local queue manager, if it does not already exist, by entering the following commands:

  ZMQSC DEF QMGR-TPFQM MAXMSGL-100000

  ZMQSC START QMGR

  Define the local queues and processes required by the WebSphere MQ communications transport and communications bindings by entering the following commands:

  ZMQSC DEF QL-'CalculatorQueue' TRIGTYPE-EVERY TRIGGER PROCESS-PCSO8

  ZMQSC DEF QL-'CalculatorSyncReplyQueue'

  ZMQSC DEF QL-'CalculatorAsyncReplyQueue' TRIGTYPE-EVERY TRIGGER PROCESS-PCCMQ

  ZMQSC DEF PROCESS-PCSO8 APPLICID-CSO8

  ZMQSC DEF PROCESS-PCCMQ APPLICID-CCMQ

- If you are using the sample User transport, ensure the necessary directories are defined by entering the following commands:

  ZFILE mkdir /usr/local/apache2/cgi-bin
  ZFILE mkdir /usr/local/apache2/cgi-bin/calcService

  Create the necessary CGI scripts in the`/usr/local/apache2/cgi-bin/calcConsumer` directory on your z/TPF system by entering the following commands:

  ZFILE echo '"#"!QWC4' > /usr/local/apache2/cgi-bin/calcService/add
  ZFILE echo '"#"!QWC4' > /usr/local/apache2/cgi-bin/calcService/sub

  Ensure the permissions are set correctly by enetering the following command:

  ZFILE chmod 755 /usr/local/apache2/cgi-bin/calcService/*

  Ensure your Apache server is configured to route the cgi requests to the correct directory by doing the following.

For Apache 1.3 and Apache 2.2 add the following line to the appropriate HTTP configuration file:
ScriptAlias /calcService/ /usr/local/apache2/cgi-bin/calcService/

Define and start the INETD listener for the Apache HTTP server. Ensure the Apache configuration includes `mod_alias` and `mod_cgi`. Please refer to the Apache readme for for information.

6.0 RUNNING

The SOAP consumer sample application demonstrates the use of the following APIs:

- `tpf_soapInitHandle`
- `tpf_soapGetOpts`
- `tpf_soapSetOpts`
- `tpf_soapInvokeService`
- `tpf_soapEnd`

The application invokes the `/CalculatorService` sample Web service on z/TPF. The `/CalculatorService` exposes two operations: `add` and `sub`. You can specify which operation to invoke, along with the two values that are to be added or subtracted. Optionally, you can specify the transport to use to send the SOAP request to the service provider. If you do not specify a transport, the first transport listed in the consumer Web service deployment descriptor will be used by default. Lastly, you may specify the message exchange pattern (MEP) to be used when processing the response. The MEP can be either synchronous or asynchronous. If synchronous is specified, the entry control block (ECB) issuing the `tpf_soapInvokeService` API will be blocked until the response message is received; if asynchronous is specified, the application will receive control immediately after the SOAP request has been sent, and a new ECB will be created to process the SOAP response.

The expected input format for the sample SOAP consumer application is as follows:

`ZCALC` *`operation`* `VAL1-`*`value1`* `VAL2-`*`value2`* `[Transport-`*`transport`*`] [Mep-`*`mep`*`] [TImeout-`*`to`*`]`

Where:

- *operation* is one of the following (case-sensitive):
  - add
  - sub
- *value1* is an integer value
- *value2* is an integer value
- *transport* is one of the following:
  - LOCAL
  - TPF
  - HTTP
  - MQ
  - USER

- *mep* is one of the following:
  - SYNC
  - ASYNC
- *to* is the timeout value in seconds

For example:

ZCALC add VAL1-100 VAL2-400 TRANSPORT-HTTP MEP-SYNC

Note: It is not valid to specify both TRANSPORT-LOCAL and MEP-ASYNC.

The sample SOAP consumer application (QWC2) will parse the input message and then do the following:

1. Initialize a SOAP consumer handle for the `/CalculatorService` Web service.
2. Determine if the TRANSPORT option was specified. If so, it will loop through the available transports specified in the handle. If the requested transport type is found, it will be marked as the active transport to be used when the service is invoked.
3. Determine if the asynchronous MEP was specified, and if so, set up the necessary data to pass to the newly created ECB upon receiving the response message.
4. Extract the values that were specified on the input command and enter them into the `t_soapParms` structure to be passed to the Web service stub to create the SOAP request message.
5. Invoke the requested operation.
6. When the response is received, process it and display the result to the operator console.
7. End the SOAP consumer handle.

7.0 NOTICES

IBM may not offer the products, services, or features discussed in this information in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. IBM may have patents or pending patent applications covering subject matter described in this information. The furnishing of this information does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 31BA/Building 008
Mail Drop P369
2455 South Road
Poughkeepsie, NY 12601-5400
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee. Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

7.1 Trademarks

IBM is a trademark of International Business Machines Corporation in the United States, other countries, or both.

Microsoft is a registered trademark of Microsoft Corporation in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.