This tutorial will introduce you to Topaz Total Test. Topaz Total Test allows for rapid creation of COBOL unit tests. Topaz Total Test can create Data stubs and Programs stubs allowing you virtualize those components of the application.
Table of Contents

SETTING UP .......................................................................................................................... 4
RUNNING THE UNIT TEST .................................................................................................. 10
REVIEWING THE UNIT TEST RESULTS .............................................................................. 20
GENERATE TEST CASE FOR SUBPROGRAM TEST ............................................................ 21
REVIEWING THE UNIT TEST RESULTS .............................................................................. 34
NOTE: if you wish to rerun this Test Drive Script, simply restart the Test Drive script from page 15. The files and libraries will be regenerated

• This tutorial will introduce you to Topaz Total Test. Topaz Total Test allows for rapid creation of COBOL unit tests. Topaz Total Test can create Data stubs and Programs stubs allowing you virtualize those components of the application.

• Topaz Total Test uses Xpediter to collect program call parameters, I/O from (QSAM, VSAM, Db2, IMS BMP, IMS MPP, CICS) as well as the results data.

• Test cases can then be run independently using stubs, removing the dependency to data/program accessibility.

Instructions:

• This guide contains many screenshots to provide a visual reference

• Every action you must take is in BOLD

• Please note each place that you must enter your own specific ID or number

• You must complete each step before proceeding to the next to successfully follow the test drive script.

You may get the following screen:

Change the User id from Photonuser to your assigned Test Drive ID (i.e. cwezxxx) and enter 123456 as the password. Click OK.
SETTING UP

Topaz Total Test Unit Test uses the Xpediter command ‘Generate Test Case’ to create the Unit test project. After creating the unit test cases, we’ll use Project Explorer to view and run the unit test case.

Open Topaz Workbench.
Open the Host Explorer perspective.
You should have already created a z/OS dataset filter for CWEZxxx.WBSAMP.*
Expand the filtered dataset list to locate and highlight CWEZxxx.WBSAMP.JCL

Locate the JCL member CWKTVSKS and right-click to display the context menu.
Highlight ‘Debug As’ for Xpediter Batch Debug Session
This will open the Debug Configuration dialog completed with the details extracted from the JCL.

Click Debug. (You may have to enter your userid and password again) you may be prompted to display the Xpediter perspective if so select ‘YES’.

The Xpediter perspective will populate several views. The main program will encounter a default breakpoint at the procedure division.
In the source code view, right-click to display the context menu and choose Generate Test Case.

The Generate Test Case dialog box will appear, next to the Project field, click New... to create a new project.
Note: The project field may or may not be populated when this dialog box is first displayed. This TT dialog is used to determine what Stubs to create as part of the Test Case.

Enter the project name My_TopazTTProject and select Finish.
The tab ‘Unit Test Options’ portion of the Generate Test Case dialog contains checked boxes that control the generation of stubs.

Program stubs allow a unit test to simulate subprogram. This is especially beneficial when the called code is incomplete and/or unavailable.

Data used during the Unit Test creation process is captured and stored with the unit test project. This allows the Unit Test to be run even if the test data has changed or is not available on the target system.

The default is set to a relatively low number such as 100, as shown here. Unit testing is designed to test with a limited number of input records compared to a full regression test that may use full production size input data.

*Verify that all stubs are selected so all relevant stubs are created, then click OK*
Click the Resume icon on the toolbar or press F8 to resume program execution.

The terminated message will appear within a couple of minutes in the Debug view.
RUNNING THE UNIT TEST

After the program terminates, open the Host Explorer perspective.

Select the Project Explorer tab. Navigate to and expand the project that was just created and expand the Unit Test folder.
Expand the JCL folder and double-click Runner.jcl to open the jcl in the editor panel.

The RUNNER JCL requires change to the job card as well as two datasets to the STEPLIB concatenation.
Rather than change the JCL, return to Host Explorer and expand the dataset list and then double click member ‘RUNNER’ in the CWEZnnn.WBSAMP.JCL

Press Ctrl + A to select all lines of member ‘CWEZnnn.WBSAMP.JCL(RUNNER)’, right click and select Copy. Return to Project Explorer click in the Runner.Jcl edit view, Press Ctrl + A, right click and select Paste. This will replace the existing lines of JCL with the lines copied from the (RUNNER) PDS member.
Close the tab by clicking on the X and select Yes to save your changes.

Expand the Stubs folder in the project.

Shown is a program stubs for the CWKTDATE module which is called from the main program. Also see two I/O stubs, one for the EMPFILE input file and one for the RPTFILE output file. If DB2, IMS or CICS where involved in the program, you would see similar stubs for the resources associated with those technologies.
Double-click on the CWKTDATE stub to open in the editor pane (the middle pane).

One row for each call to CWXTDATEn from the main program showing data passed to the sub program via the linkage section is shown.

Double-click on the EMPFILE stub to open in the editor pane (the middle pane).

One row for each record read from the EMPFILE dataset during the Unit Test creation is shown. The Value column displays and allows you to change the values in the input record for future runs of the Unit Test.
Double-click on the RPTFILE stub to open in the editor pane (the middle pane).

One row for each record written to the RPTFILE dataset during the Unit Test creation is shown. The Comparison and Expected Value columns are used to validate that the program writes the correct information when the Unit Test is run.

Expand the Scenarios folder and double click on the CWKTVSKS.Scenario.testscenario to open it in the editor panel.
The Input Data tab at the top will show the parms passed to the program.

The Stubs tab allows you to include or exclude which stubs will be used when the Unit Test is run. If DB2, IMS or CICS stubs are present, all those stubs must be used.
The Properties tab allows you to document notes and additional information pertaining to the Unit Test Scenario.

Click on the Run Test Scenario icon on the toolbar. If you get the Logon to Repository see page 3.
In the Test Run dialog box that appears, Select Test Drive in the Host pull down menu, Select Unit Test tab.

Click ‘Use Jcl Defined in this Project’. Then click OK.
You may see a status box like the one shown, which will show the progress of the Unit Test execution. Otherwise you can see progress on the lower right of the screen:

```
Running test suite: CWKTVnario_testsuite
```

REVIEWING THE UNIT TEST RESULTS

When the Unit Test completes, the report will be displayed in the Editor panel. This report will show a Success rate, number of failures and errors and where the failures occurred.

This concludes the Automated Unit testing using Topaz for Total Test.

Close all of the open tabs in the editor pane.
GENERATE TEST CASE FOR SUBPROGRAM TEST

Topaz for Total Test allows you to generate a unit test for testing your subprograms. First, we'll use the Xpediter debug configuration used for creating our Test Case for the CWKTVSKS program. After setting up the unit tests, we'll use Project Explorer to run them.

Open the Run dropdown from the top navigation, and choose Debug Configurations...
Select ‘CWKTVSKS’ or the name of the debug configuration you defined for Unit Test creation in previous section. You will use the same Debug configuration as used for the Unit Test Case creation. You will still need to invoke the full application to generate the Test Case for the Subroutine.

Click Debug. (You may have to enter your userid and password again). Xpediter will populate numerous views as the program runs. The main program will encounter a breakpoint at the procedure division.

Once in the Xpediter Perspective enter the Main program CWKTVSKS name in the DDIO view. This will list any of the CSECTs (subprograms) associated with the main program.

Right click Subprogram CWKTDATE and select Set Initial Breakpoint
You will now see both the Main and subprogram source listings.

Click the Resume icon on the toolbar or press F8
Notice that Xpediter is paused at the start of the Subprogram

In the source code view, right-click to display the context menu and choose Generate Test Case.
Best Practices suggest that a New Project be created for the new Test case.
The tab ‘Unit Test Options’ portion of the Generate Test Case dialog contains checked boxes that control the generation of stubs.

If your Subprogram performs calls to additional subprograms then Program stubs will be created.

I/O stubs allow a unit test to bypass reads and writes. Data used during the Unit Test creation process is captured and stored with the unit test project. This allows the Unit Test to be run even if the test data has changed or is not available on the target system.

The maximum number of records should be set to a relatively low number such as 100, as shown here. Unit testing is designed to test with a limited number of input records compared to a full regression test that may use full production size input data.

Verify that all stubs are selected so all relevant stubs are created, then click OK.
Before clicking the Resume icon (F8) on the toolbar, uncheck the CWXDATE breakpoint in the Breakpoints view, this subroutine gets called several times.

Click Resume or F8 (if not already done so).

Once test is complete switch to “Host Explorer” perspective and click on “Project Explorer” view.
Since the JCL modifications are the same as the previously created test case, you can copy/paste the Runner JCL from one project to another.

Select Runner.jcl from the first project, right click and select Copy.

Select the JCL Folder in the new project, right click and select Paste.
Click “yes to all”.

Expand the Scenarios folder and double click on the CWKDATE scenario to open it in the editor pane.
The Input Data tab at the top shows the data that passed to the subprogram via the linkage section. These values can be changed to meet the unit test requirements.

The Stubs tab doesn't show any stubs because this program didn’t call any additional programs or perform any I/O.
The Assertion tab shows what fields will be validated when the test case is replayed.

Click on the Run Test Scenario icon on the toolbar.
In the Test Run dialog box that appears, Select Test Drive in the Host pull down menu, Select Unit Test Tab and click OK

Select ‘Use JCL defined in this Project’ then click OK
You may see a status box like the one shown which will show the progress of the Unit Test execution. Otherwise you can see progress on the lower right of the screen:
REVIEWING THE UNIT TEST RESULTS

When the Unit Test completes, the report will be displayed in the Editor panel. This report will show a Success rate, number of failures and errors and where the failures occurred.

This concludes the Automated Unit testing using Topaz for Total Test.

NEXT STEPS
- Should you wish to restart this Test Drive script, follow the restart instructions at the beginning of this script.
- For more information about “Mainstreaming the Mainframe” visit our Mainframe DevOps site at https://devops.api.compuware.com/
- Click on the link below to return to the Compuware Test Drive main page and choose your next road trip!
- COMPUWARE TEST DRIVE