|  |  |
| --- | --- |
|  | PANTHER  WEB  DEBUGGER |

Author: Subhan K & Moulika N

Version 2.0

3 October 2017

**Installing and configuring Panther/Web Debugger**

Prerequisite

Panther Web software & a Panther/Web application to debug.

Download

Download Panther Web Debugger from Prolific FTP site.

Eclipse & Tomcat

* Install a later version of Eclipse.
* Also setup Tomcat as a Test Server in Eclipse UI. See <http://crunchify.com/step-by-step-guide-to-setup-and-install-apache-tomcat-server-in-eclipse-development-environment-ide/>
* Add web module to your tomcat server from the modules tab

JPL

* To enable tracing execute this in your very first JPL screen of your application

**call sm\_trace ("ALL websocket=default")**

JAR files to Panther/Web ini file

       Add in the CLASSPATH

* prowebdbg.jar (provided on ftp site)
* tyrus-standalone-client-1.13.1.jar
* websocket-api.jar
* javax.json-api-1.0.jar
* javax.json-1.0.4.jar
* Pro5.jar (provided in your (panther/config) folder

Adjustments to Panther/Web ini file

* WebSocketClass= com.prolifics.websocket.DebugMessenger
* NumServers=1
* IdleServerTimeout= <set to blank>
* ServerTimeout= <set to blank>

Import Debugger

* Import Gen\*. zip project into your workspace:
* File->Import...
* General->Existing Projects into Workspace
* Next
* Choose the 'Select archive file' radio button
* Browse...
* Finish
* After importing the project, edit the Libraries used in Java Build Path by replacing the locations given with the locations where those JAR files exist on your machine
* Verify that the code can be rebuilt by cleaning the project and rebuilding
* Export the project as a java jar file to the location where your **prowebdbg.jar** is present and replace that jar with your project.
* Add Web Module Gen\* to Tomcat Server

Test

Localhost: 8085/Gen\*/Debugger

Add the applications’ URL to the  Application URL field, .i.e. <http://localhost/icms/icms.jam> , then press Go.

**How to use the debugger**

Windows

**Status Window**

Here you would see the status of the Trace which gets loaded and all the JPL code which runs at the back end of the screens which are in your application.

**Breakpoint Window**

Here we can set break points based on the type which you need with respect to the line numbers and proc names as well.

**Data Watch Window**

This helps us to see any of the variable values which we have in our code.

Buttons

**Play (**Continue to next break point**)**

This helps us for continuing to the next break point if we have break points more than 1 and also it helps us to take the trace to the end of the JPL file which we are in.

**Step In (**continue to next event**)**

This button helps us to continue to the next line of the trace if we are in a particular break point and we need to check our JPL line by line.

**Step Over (**Steps over a proc**)**

This helps us in stepping over a particular proc which we might have provided as a break point or else when we step in through a JPL line where we have a proc start and we want the trace to step over at the end of that proc.

**Break (Breaks at next event)**

This button helps to break our trace where ever and whenever needed.

* If you do not have any breakpoints and you wish to check the code line by line by using **Step** **in** button (2nd from the left), you can click the **break** **button** and perform the panther event which you wish to, by which the **3** buttons (**Play, Step In and Step over**) are enabled from there you can do either of the things like below.
  + Step In line by line
  + Step Over at the end of proc if a proc has started

Start/Stop Tracing Debugger (Toggle debug mode)

We can click the last button with a **Bug** on it if we need to **start\stop**. By default, the trace is on which we can see that by its colour **Green** and if we need to **stop** the trace we can click the button and the button changes its colour to **Red** which indicates that the Trace is **stopped**.

How to set a Breakpoint

* Select the **type** by which you need to set a break point
* If the type you choose is **Any Jpl** you have to provide a **proc** name at the **location** which is in your JPL file and you can proceed with the "**Play Button** or **Step over** button" to continue further
* If the type you choose is **JPL Module** you have to provide the **JPL** file name for which you want to set a break point.
* At the **location** you need to enter your **proc** name If you want to set break point for a **proc**.
* If you need to set a break point for a **line** number, you should enter a **line** **number** and Press **enter**.

There by you can proceed further by clicking the **Play** button at the first or else **Step** **over** button to **continue** **trace**.

How to watch Data

In this section you need to enter the name of your variable which you wish to see the value.

* Set a break point, enter a variable name in the “variables & Expressions” tab in Data watch and click the Play button.
* Tracing will be stopped at breakpoint then click step-in button which is 2nd from the left.
* This button shows up the trace line by line and the moment where your variable is assigned with a value in your JPL that particular value will be showed up on the Data watch window if no value it shows up a question mark “?”.

NOTE:

\*\*Once you set a Breakpoint you must continue Debugging; i.e. hit Play/Step Over. You cannot manipulate the UI

Breakpoints are NOT saved when you refresh screens