PANTHER WEB DEBUGGER

Panther's Web debugger provides you the ability to view and analyze all JPL source in a Panther Web application. You can:

- Step through JPL, stopping to examine data.
- Set breakpoints in JPL code on which to interrupt program execution.
- Examine application data in JPL variables, JPL arrays or expressions.
- Review tracing activity in the log file. Tracing messages can also be written to a user-specified log file.

Installing and configuring Panther/Web Debugger Win32

Prerequisite

A Panther/Web application to debug, Eclipse and Apache

Download

Download Panther Web Debugger from https://github.com/ProlificsPanther

Eclipse & Apache Tomcat

i. Install a later version of Eclipse- 32bit
ii. Please setup Apache Tomcat as a Test Server in Eclipse UI. Click for steps.

Add JAR files to CLASSPATH in Panther/Web ini file

i. prowebdbg.jar
ii. tyrus-standalone-client-1.13.1.jar
iii. websocket-api.jar
iv. javax.json-api-1.0.jar
v. javax.json-1.0.4.jar
vi. pro5.jar. See use the exact version from $SMBASE/config

Update Panther/Web ini file

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

Enable tracing in your application

To enable tracing, execute the below call in your very first JPL screen of your application, or in a JPL file that is loaded when your application first starts, .i.e. file point to by SMINITJPL

```
call sm_trace("ALL websocket=default")
```

Import Debugger zip File

1. Import Gen*. zip project into your workspace:
2. File->Import...
3. General->Existing Projects into Workspace
4. Next
5. Choose the 'Select archive file' radio button
6. Browse...
7. Finish

Fix Java Build Path and export prowebdbg.jar

1. After importing the project, edit the Libraries used in Java Build Path by correcting the locations of the JAR files on your machine. Right click on the Project -> Properties->Java Build Path
2. Verify that the code can be rebuilt by cleaning the project and rebuilding
3. Export the project as a java jar file to the location where your **prowebdbg.jar** is present and replace that jar with your project.
4. Add Web Module Gen* to Tomcat Server

*If you have conflicting port numbers, please chose an available port number.

Test the Debugger

1. Enter this URL to access the Debugger http://localhost: 8085/Gen2a/Debugger
2. Next add the applications' URL to the Application URL field, .i.e. http://localhost/hr/login.jam , then press Go.
3. The Application will open in the next Browser Tab

How to use the debugger

Windows

Status Window
Contains all the trace details, events, JPL, line numbers etc. This is what the JServer processes.

Breakpoint Window
Choose the breakpoint type here.

Data Watch Window
Ability to introspect values of variable/fields.

JPL Module Window
In this window you will be able to see the name of the **JPL** file that us executing. When **breakpoints** are set, you can navigate to this **window** and check to see which **JPL** file is being executed.

Buttons

Play (Continue to next break point**)**
As the name indicates this button helps us continue to the **next breakpoint**.

- For example, if we have set **3 breakpoints,** once we start **debugging,** it will **stop** at the **1st breakpoint**; if we want to inspect the **JPL** line by line, select the **step in** button (**2nd from Left**) OR use the **Play** button to continue to the **next break point** ignoring the remaining JPL code.

- If only **1 breakpoint** is set and when it stops at that **breakpoint**, if we choose **Play,** it takes us to the end of that **JPL** file, which specifies that we don't have any further **breakpoints** set.

Step In (Continue to next event**)**
Allows you to debug JPL within a particular JPL procedure that you have previously set as a breakpoint.

Step Over (Steps over a proc**)**
Let's you step over a particular **JPL** procedure that was set as a breakpoint.

Break (Breaks at next event)
This button helps to break our trace where ever and whenever needed.

- If you have not set any breakpoints and you wish to check the code line by line using the **Step in** button (2nd from the left), you can click the **Break** button and perform any Panther event. The **Play, Step In and Step over** buttons are enabled from there and you can do either of the 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)

The button with the Bug image allows you to **start\stop** tracing. By default, tracing is turned on; i.e. Bug is **Green**. To stop tracing, toggle the button and it will change to Red which indicates that tracing has **stopped**.

Setting Breakpoints

- Select the breakpoint **type** for which you need to set a breakpoint
- If the type you choose is **Any Jpl,** provide a **proc** name in the **location** field, which is the name of your **JPL** procedure. Then you can proceed with the "**Play Button** or **Step over** button" to continue further.
- If the type you choose is a **JPL Module** you will have to provide the **JPL** file name for which you want to set a breakpoint in.
    - If you want to set a breakpoint in a **proc**, enter a JPL **proc** name in the **location** field
- If you need to set a breakpoint for a line number please enter a **line number** and Press **enter**.
- To continue debugging, click on the **Play** or **Step Over** button.

Watching Data

In this section you need to enter the name of a field/variable that you like to view its value.

- Set a breakpoint, enter a variable name in the “Variables & Expressions” tab and click the **Play** button.
- Tracing will stop at the breakpoint. Click the **Step In** button. As you trace the JPL, you will be able to view the value of that variable in this window.
- A “?” value indicates that the variable/jpl has no value.

NOTE:

**Once you set a Breakpoint you must continue Debugging; i.e. hit Play/Step Over.
Breakpoints are NOT saved when you refresh the browser.