8376
Java Application Development using Eclipse Part 2

Jezz Kelway – kelwayj@uk.ibm.com
Java Technology Centre, z/OS Service
IBM Hursley Park Labs, United Kingdom

Objectives

- Presentation 1
  - An overview of Eclipse and its Architecture
- Presentation 2
  - The Debugger
  - Plugin's

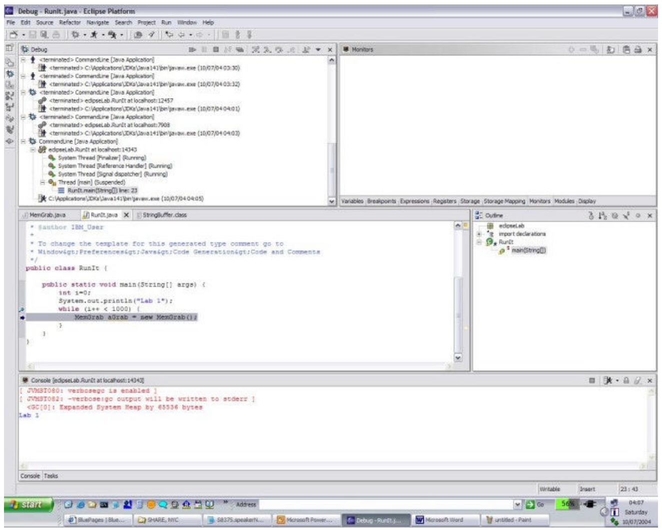
Agenda

- The Debug Perspective
  - Local debugging
  - Remote debugging
- Plugin’s
  - Description
  - Demonstration

Debug Perspective

- The debug perspective gives you access to functionality which enables the user to run code interactively by:
  1. Stepping through the execution line by line.
  2. Setting break points at places in the code where the execution will suspend
  3. Examining program attributes, such as variables, locks, memory, registers,... at any given (break) point
  4. Modifying variables during program execution.
  5. Modifying code during program execution
- This enables the user to:
  1. Find code errors.
  2. Unit test

Debug Perspective



Debug - RunIt.java - Eclipse Platform

```
<terminated> CommandLine [Java Application]
    <terminated> C:\Applications\JDKs\Java141\bin\javaw.exe (10/07/04 03:30)
<terminated> CommandLine [Java Application]
    <terminated> C:\Applications\JDKs\Java141\bin\javaw.exe (10/07/04 03:32)
<terminated> CommandLine [Java Application]
    <terminated> eclipseLab.RunIt at localhost:12457
    <terminated> C:\Applications\JDKs\Java141\bin\javaw.exe (10/07/04 04:01)
<terminated> CommandLine [Java Application]
    <terminated> eclipseLab.RunIt at localhost:7908
    <terminated> C:\Applications\JDKs\Java141\bin\javaw.exe (10/07/04 04:03)
CommandLine [Java Application]
    eclipseLab.RunIt at localhost:14343
        System Thread [Finalizer] (Running)
        System Thread [Reference Handler] (Running)
        System Thread [Signal dispatcher] (Running)
        Thread [main] (Suspended)
            RunIt.main(String[]) line: 23
    C:\Applications\JDKs\Java141\bin\javaw.exe (10/07/04 04:05)
```

```java
 * @author IBM_User
 *
 * To change the template for this generated type comment go to
 * Window&gt;Preferences&gt;Java&gt;Code Generation&gt;Code and Comments
 */
public class RunIt {

    public static void main(String[] args) {
        int i=0;
        System.out.println("Lab 1");
        while (i++ < 1000) {
            MemGrab aGrab = new MemGrab();
        }
    }
}
```

Console [eclipseLab.RunIt at localhost:14343]

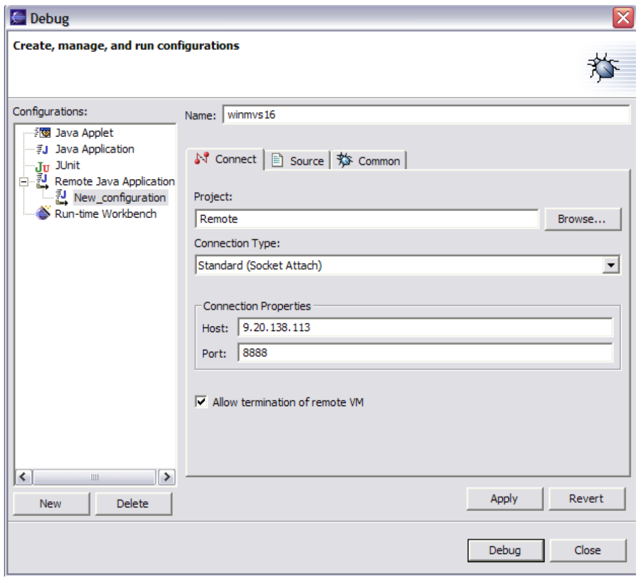
```
[ JVMST080: verbosegc is enabled ]
[ JVMST082: -verbose:gc output will be written to stderr ]
  <GC[0]: Expanded System Heap by 65536 bytes
Lab 1
```

File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Monitors
Variables Breakpoints Expressions Registers Storage Storage Mapping Monitors Modules Display
MemGrab.java
RunIt.java
StringBuffer.class
Outline
eclipseLab
import declarations
RunIt
main(String[])
Console Tasks
Writable
Insert
23 : 43

Remote debugging

- Debug a java application running on z/OS from Eclipse running on a PC.
- Configure Eclipse
  - Build the class to be debugged in a project as before.
  - The source code is the same as on the remote machine
  - Select Run>Debug>Remote Java Application and create a configuration
  - Fill in host (IP address) / port (where the remote VM is accepting connections) and source details, see next slide

Remote debugging, Eclipse side


Debug

Create, manage, and run configurations

Configurations:
- Java Applet
- Java Application
- JUnit
- Remote Java Application
  - New_configuration
- Run-time Workbench

Name: winmvs16

Connect | Source | Common

Project: Remote

Connection Type: Standard (Socket Attach)

Connection Properties
Host: 9.20.138.113
Port: 8888

[✓] Allow termination of remote VM

Remote debugging, remote side

- Build the java application with the –g flag

  Javac –g Polynomial.java

- Start the JVM in server/suspend mode, so that you can then attach the Eclipse Java debugger to it.

  The following is the content of a script 'runMe'

```
java -Xdebug -Xnoagent -Djava.compiler=NONE -runjdwp:transport=dt_socket,server
=y,suspend=y,address=8888 $* &
```

Remote debugging, pulling it together

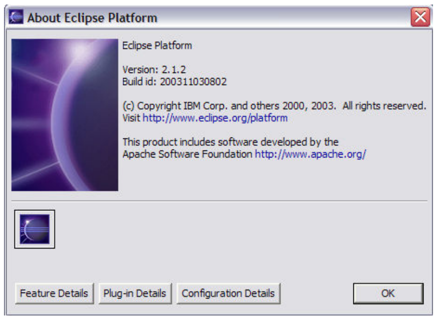
- Start the java application on the remote machine with ‘runMe Polynomial’. Polynomial is being executed but is suspended listening on port 8888
- Start the remote debug session in Eclipse. Step through the application and watch the output appear in the remote console.

Lab 2 summary

- Setting breakpoints to control program execution
  - Stepping through line by line
  - Entering and leaving functions
  - Resume execution to next breakpoint
  - Conditional breakpoints
- The following can be viewed during execution
  - Variables
  - Thread stacks
- The following can be changed during execution
  - Variables
  - Code (Hot Swapping)

Plug-in introduction

- Eclipse is a framework which enables plug-in operability and interoperability
- Click on ‘Help>About Eclipse Platform’ then ‘Plug-in Details’ to see currently active Plug-ins
- http://www.eclipse.org/documentation/html/plugins/org.eclipse.platform.doc.isv/


About Eclipse Platform

Eclipse Platform

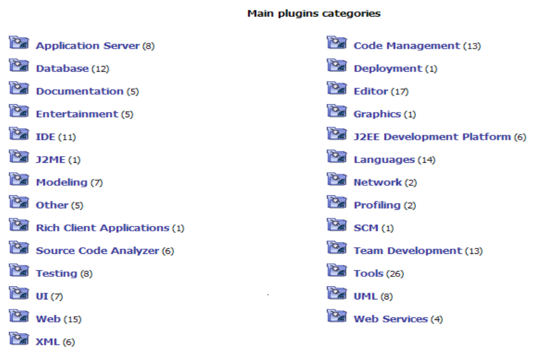
Version: 2.1.2
Build id: 200311030802

(c) Copyright IBM Corp. and others 2000, 2003. All rights reserved.
Visit http://www.eclipse.org/platform

This product includes software developed by the Apache Software Foundation http://www.apache.org/

Where to get Plug-ins

- Eclipse plug-in download site http://www.eclipseplugincentral.com/Web_Links+main.html


Main plugins categories

Application Server (8)	Code Management (13)
Database (12)	Deployment (1)
Documentation (5)	Editor (17)
Entertainment (5)	Graphics (1)
IDE (11)	J2EE Development Platform (6)
J2ME (1)	Languages (14)
Modeling (7)	Network (2)
Other (5)	Profiling (2)
Rich Client Applications (1)	SCM (1)
Source Code Analyzer (6)	Team Development (13)
Testing (8)	Tools (26)
UI (7)	UML (8)
Web (15)	Web Services (4)
XML (6)	

- Also; http://www.eclipse-plugins.com/

Presentation review

- 'Ease-of-use' features such as code generation, automatic syntactic checking and code completion reduce the time needed to write software
- Debugging options including, step through execution, breakpoints, display of variable values, memory, registers, ... make finding those algorithmic bugs easier
- Remote debugging facilities will leave you open mouthed.
- Plugin’s, if you need something specific, write a plug in !