8358
Java Application Development using Eclipse

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

Abstract

- Learn how to use the powerful features of Eclipse to aid software development.
  - '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.

Objectives

- An overview of Eclipse and its Architecture
- The Debugger
- Plugin’s

Agenda

- Introduction
  - Eclipse Overview and History
  - Eclipse Architecture
  - Perspectives

- The Resource Perspective
  - Editing and syntax checking
  - Running Programs

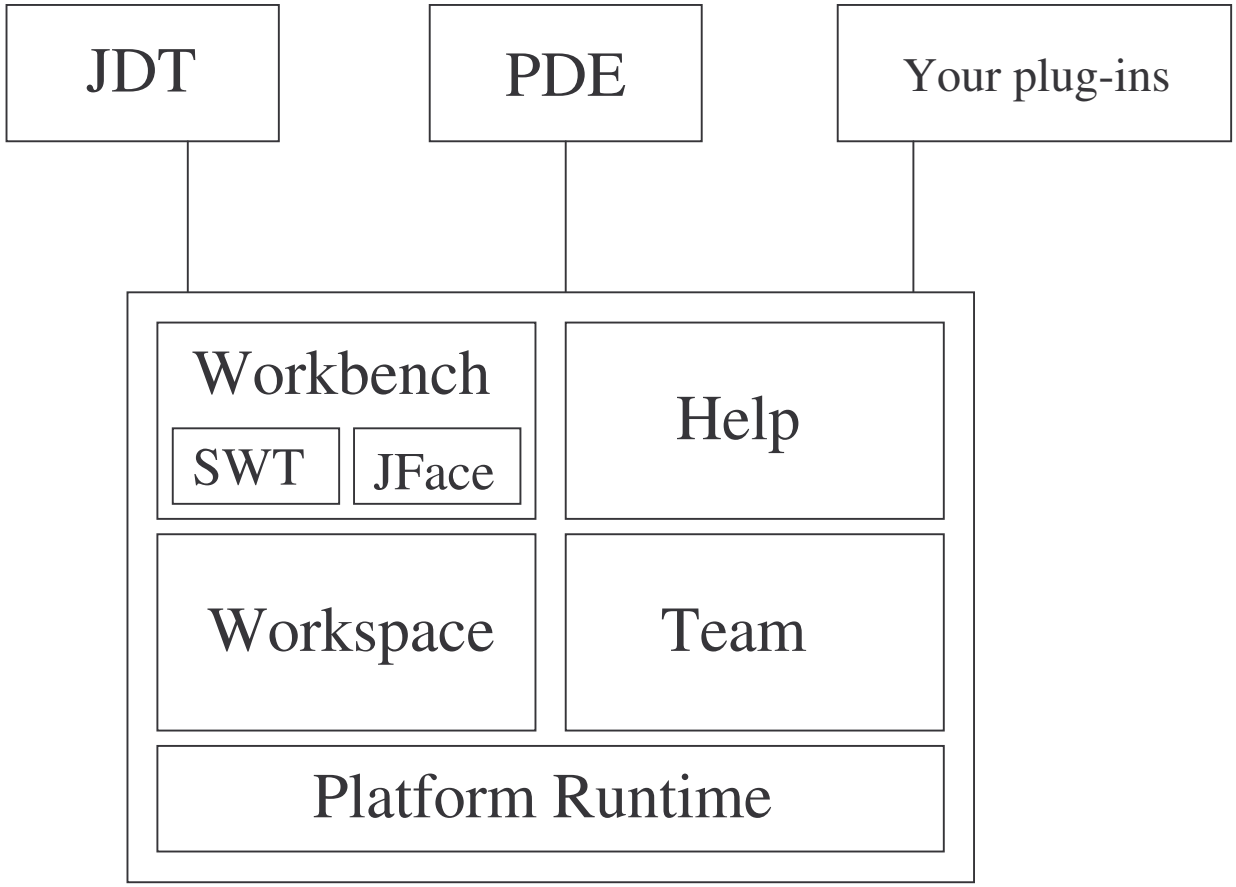
- The Debug Perspective
  - Local Debugging
  - Remote Debugging

- Plug-In's

Eclipse Overview and History

- Eclipse is an extensible software development framework based on a plug-in architecture.
- Eclipse is open source, released under a common public license (CPL).
- Originally developed by IBM, version 1.0 being released in November 2001, Eclipse is now under the stewardship of an independent consortium including: IBM, SAP, Sybase, Fujitsu, HP, BEA…

Eclipse Architecture



Eclipse Architecture explained

Platform runtime

Primary job is to discover the available plug-ins. Each plug-in has an XML manifest file which lists the connections the plug-in requires.

The workspace

Manages the user's resources which are organised into projects. The workspace maintains a history of changes to resources.

The workbench

Eclipse's Graphical User Interface, menus, toolbars and perspectives.Perspectives provide a GUI for specific areas of functionality such as debugging, plug-in development,...

The Standard Widget Toolkit (SWT) are graphics toolkits which map directly to the OS native graphics.

JFace is a toolkit built on SWT.

Team Support

Version control. Eclipse provides a client for Concurrent Version System (CVS)

Help

An extensible help component

Java Development Toolkit (JDT)

Toolkit for the writing and debugging of Java programs. C Development Toolkit is the equivalent plug-in for C development.

Plug-in Development Environment (PDE)

Developing plug-ins for extending ECLIPSE.

Perspectives

- An Eclipse perspective is a pre-selected set of views arranged in a predetermined way.
- These presentations are structured around describing functionality of the following perspectives:
  - Resource
  - Debug

Agenda

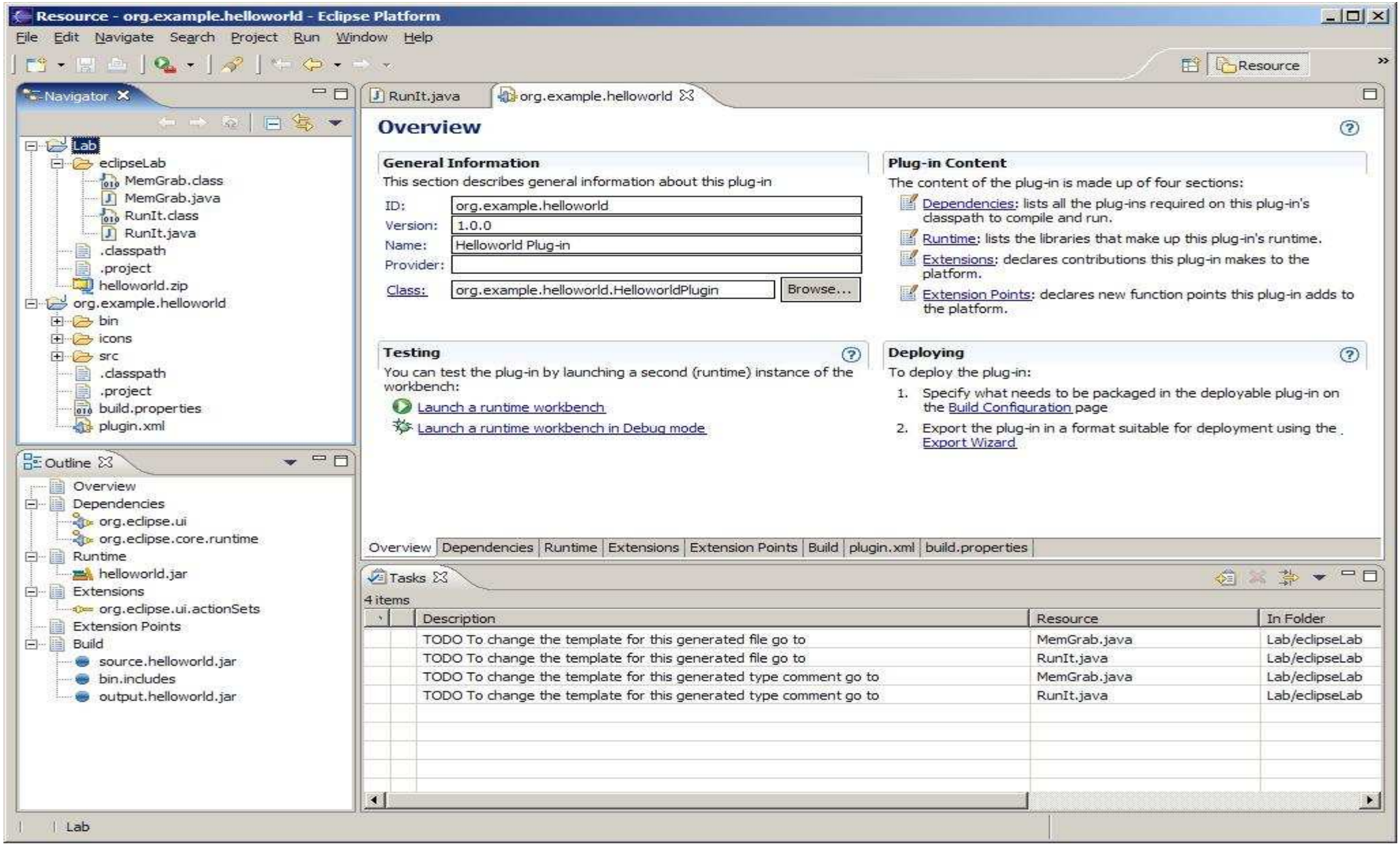
- Introduction
  - Eclipse Overview and History
  - Eclipse Architecture
  - Perspectives

- The Resource Perspective
  - Editing and syntax checking
  - Running Programs

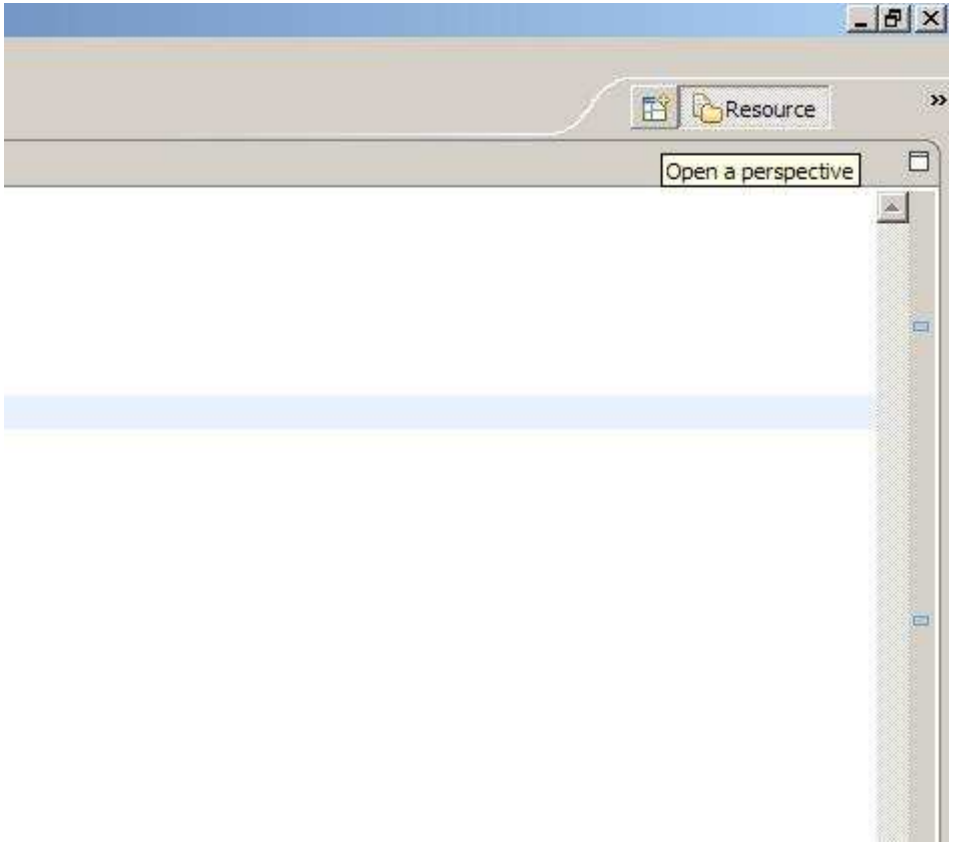
- The Debug Perspective
  - Local Debugging
  - Remote Debugging

- Plug-In's

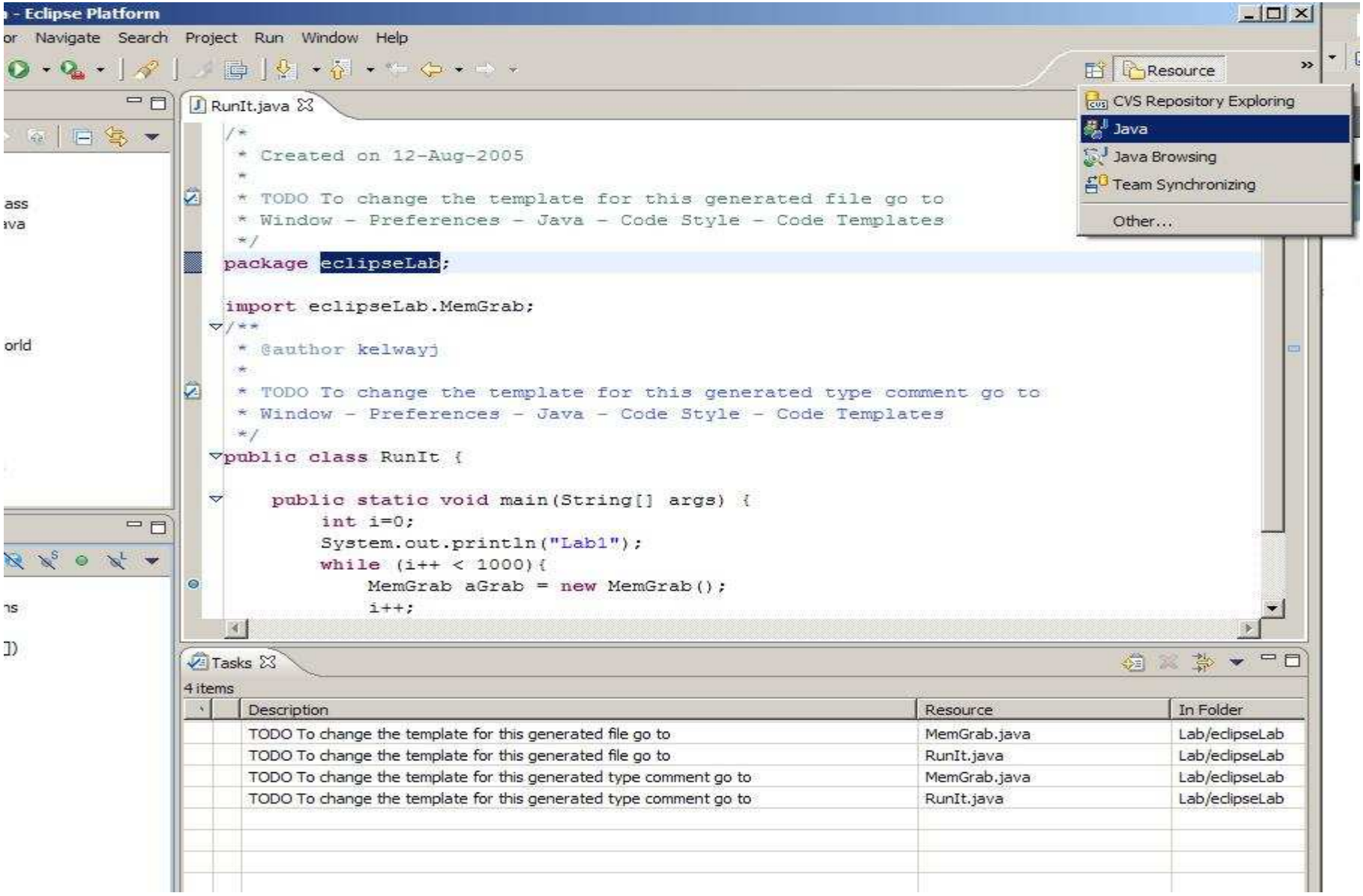
Perspectives and Eclipse 1



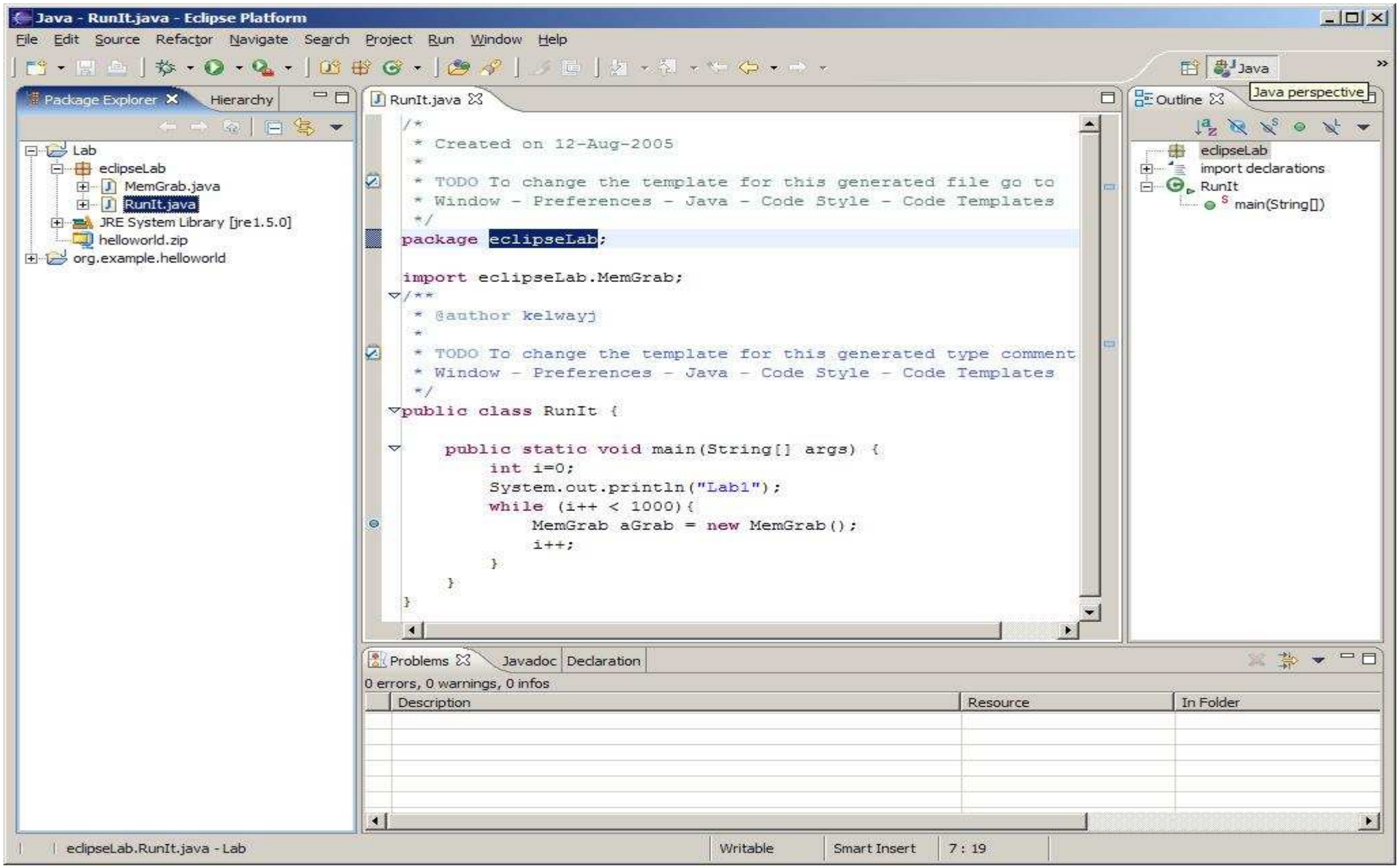
Perspectives and Eclipse 2



Perspectives and Eclipse 3



Resource perspective



Agenda

- Introduction
  - Eclipse Overview and History
  - Eclipse Architecture
  - Perspectives

- The Resource Perspective
  - Editing and syntax checking
  - Running Programs

- The Debug Perspective
  - Local Debugging
  - Remote Debugging

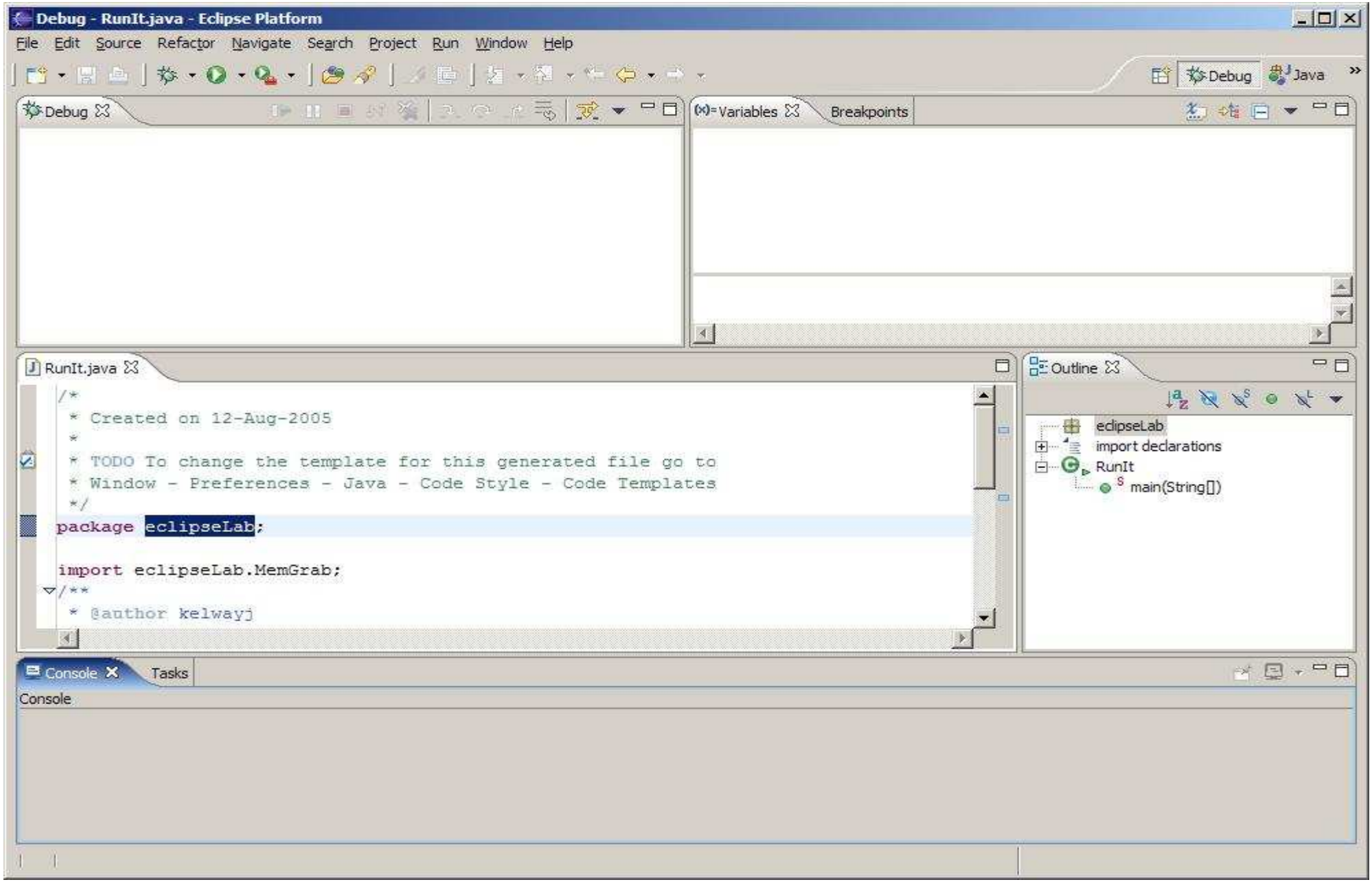
- Plug-In's

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



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, 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.

Agenda

- Introduction
  - Eclipse Overview and History
  - Eclipse Architecture
  - Perspectives

- The Resource Perspective
  - Editing and syntax checking
  - Running Programs

- The Debug Perspective
  - Local Debugging
  - Remote Debugging

- Plug-In's

Plug-In's

- 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/

Where to get Plug-ins

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

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

Lab Exercises

- Lab 1 The Resource Perspective
- Lab 2 The Debug Perspective
- Lab 3 Creating a HelloWorld Plug-In
- Lab 4 Deploying the HelloWorld Plug-In
- Lab 5