# Debugging

Dwight Deugo (dwight@espirity.com)
Nesa Matic (nesa@espirity.com)

# Additional Contributors

- None as of August, 2004

# Module Road Map

## 1. Eclipse Debugging

# Module Road Map

1. **Eclipse Debugging**
   - **Debug Perspective**
   - **Debug Session**
   - **Breakpoint**
   - **Debug Views**
   - **Breakpoint Types**
   - **Evaluating and displaying expressions**

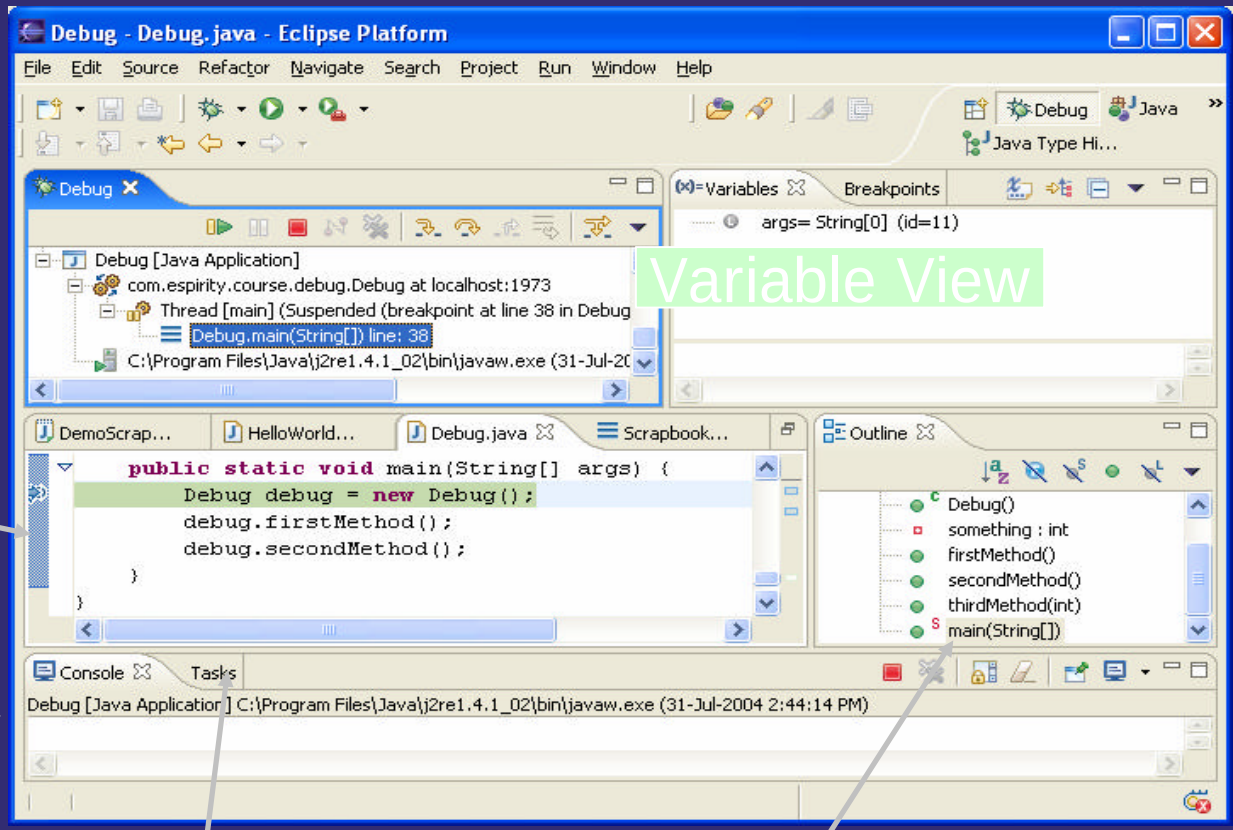
# Debugging in Eclipse

- The Java Debugger
  - Part of Eclipse Java Development Tools (JDT)
  - More than `System.out.printn("error")`
  - Detects errors as code executes
  - Correct errors as code executes
  - Actions you can perform debugging include:
    - Control Execution
    - Set simple breakpoints
    - Set conditional breakpoints
    - Review and change variable values
    - Hot code replace

# Debug Perspective

Threads and Monitor View




Editor View

Console View

Tasks View

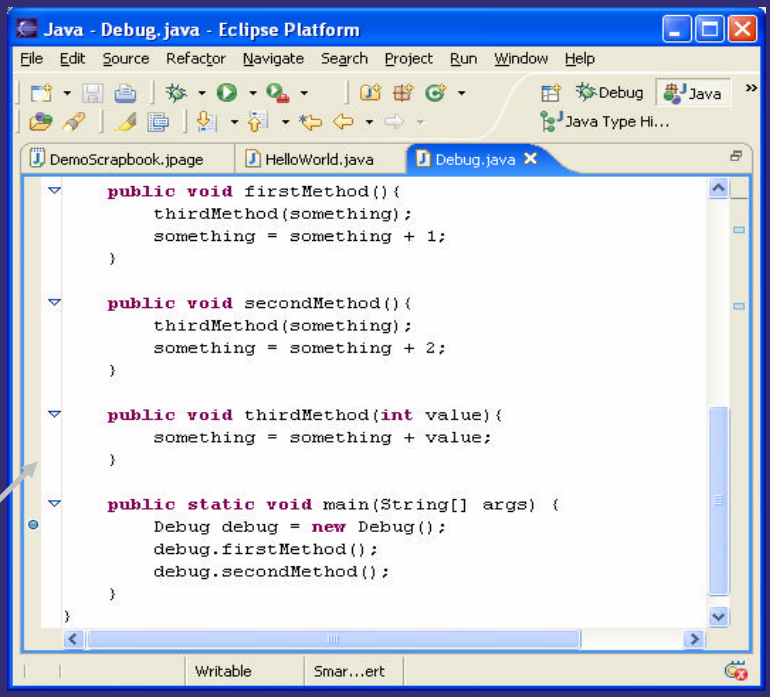
Outline View



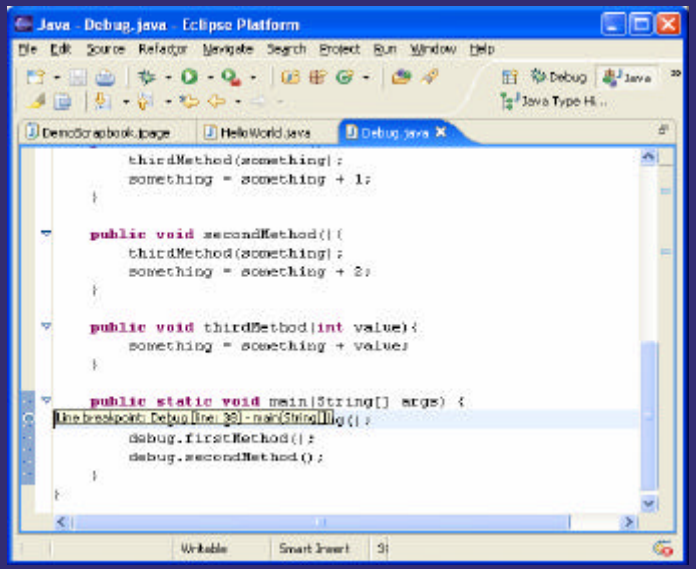
# Simple Breakpoint

- Breakpoint
  - Stops the execution of a program at the point
  - Thread suspends at the location where the breakpoint is set
- Setting a breakpoint
  - CTRL+Shift+B at current point in editor line
  - Double click in editors marker bar a current line



# Deleting Breakpoints

- Double click on the breakpoint in the editor
- CTRL+Shift+B at current point in editor line

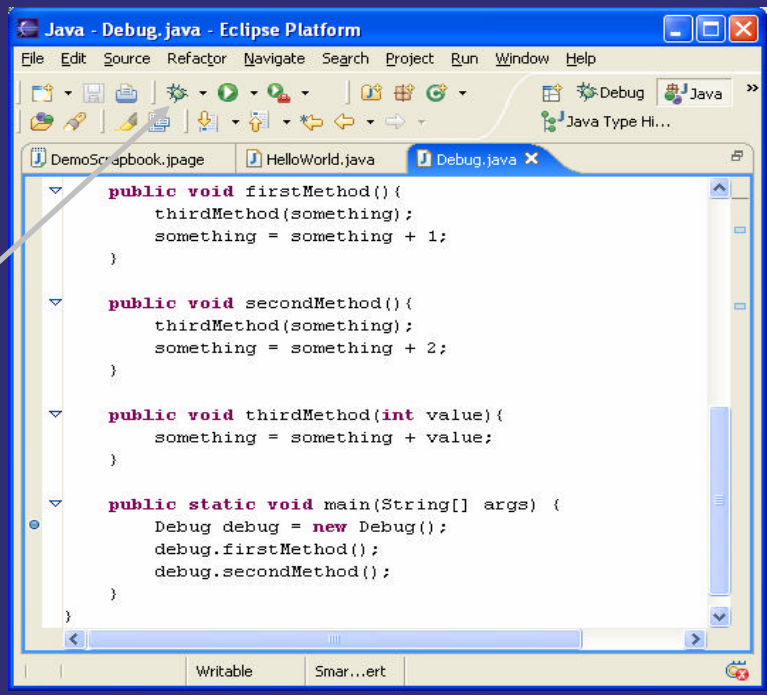




# Starting a Debugging Session

- Select Java class containing the following:
  - main() method
  - Resulting execution will pass breakpoint
- Select Run → Debug As… → Java Application
- Or Select Debug As… → Java Application from the debug menu

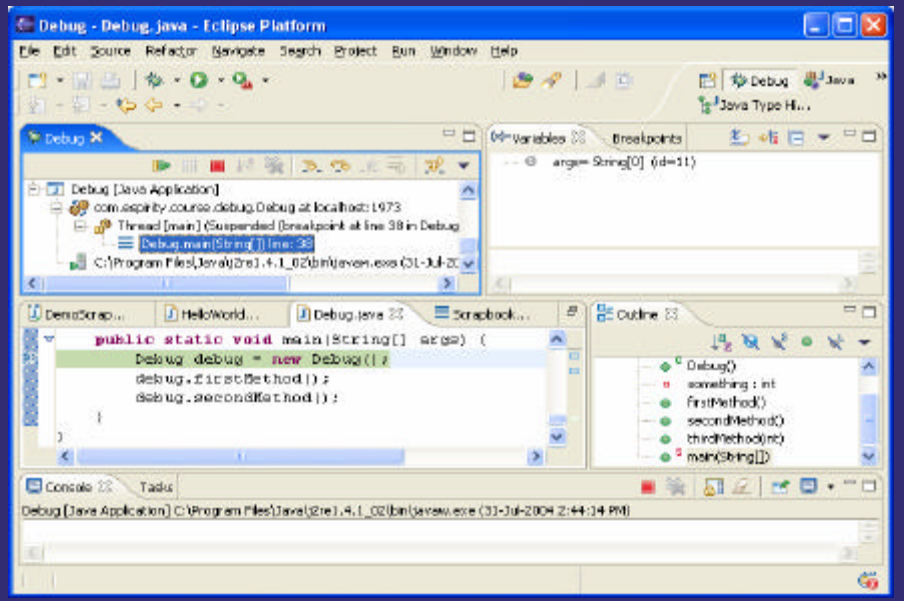




# Debug Session

- Execution suspends prior to the line with a breakpoint
- You can set multiple breakpoints

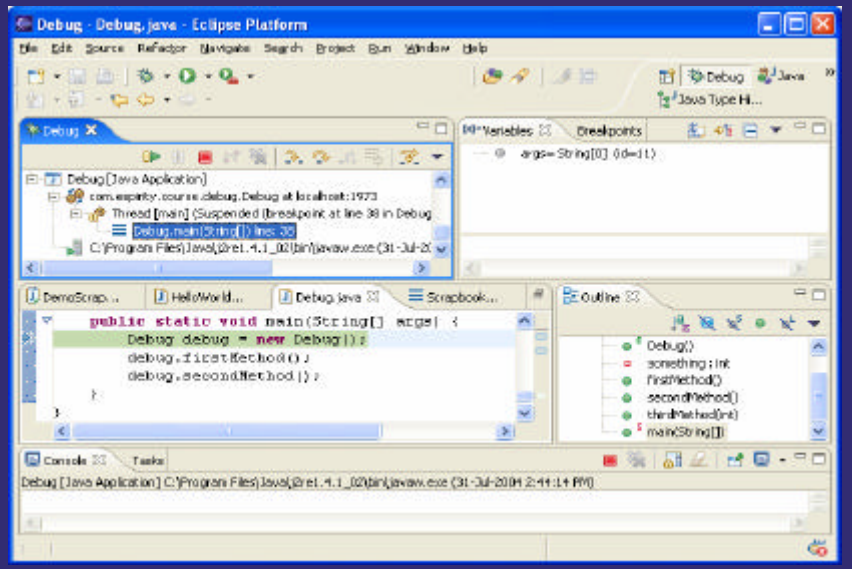




# Control Execution From Breakpoint…

- Step Into or F5:
  - For methods, execute method and suspend on first statement in the method
  - For assignments, similar to Step Over
  - For conditionals, similar to Step Over
- Step Over or F6
  - Execute next statement
- Step Return or F7
  - Resume execution to the end of the method on the next line after it was invoked

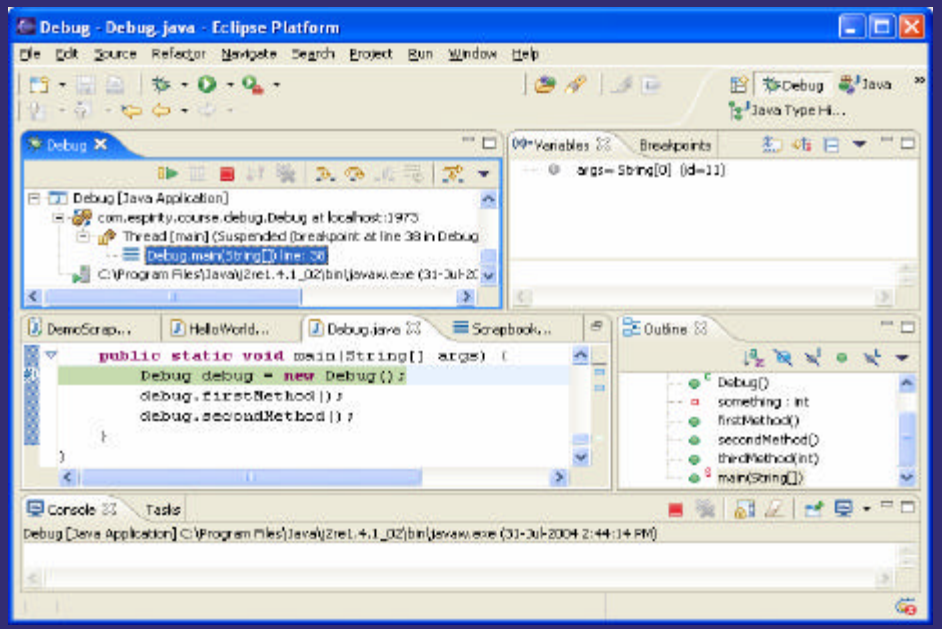




# …Control Execution From Breakpoint

- Resume or F8
  - Continue execution until program ends or another breakpoint is reached
- Terminate
  - Stops the current execution thread

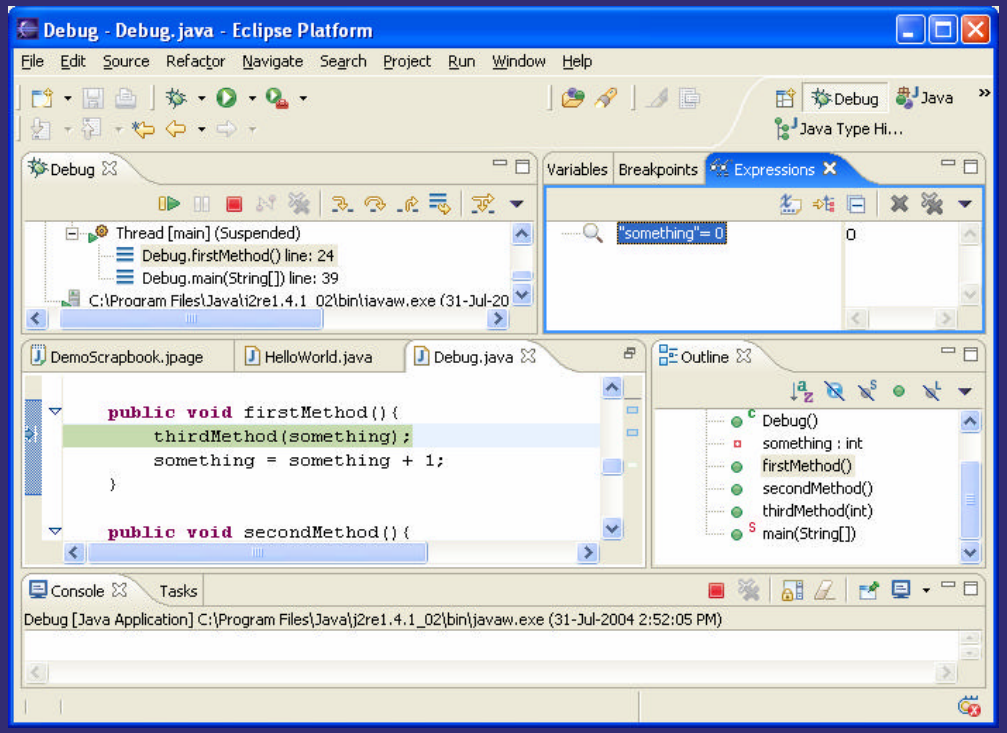




# Variables and Fields

- To see the values bound to fields:
  - Use Variables View
  - Select variable in editor and select Inspect
  - Select variable in editor and select Display





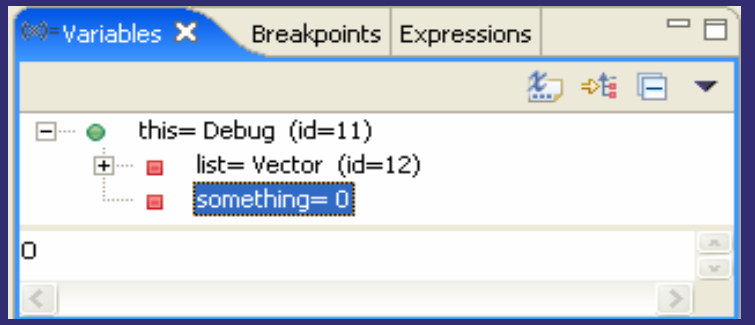
# Code Debugging in this Module

```
public class Debug {
   private int something = 0;
   private Vector list = new Vector();

   public void firstMethod(){
        thirdMethod(something);
        something = something + 1;
   }
   public void secondMethod(){
        thirdMethod(something);
        something = something + 2;
   }
   public void thirdMethod(int value){
        something = something + value;
   }

   public static void main(String[] args) {
        Debug debug = new Debug();
        debug.firstMethod();
        debug.secondMethod();}
}
```

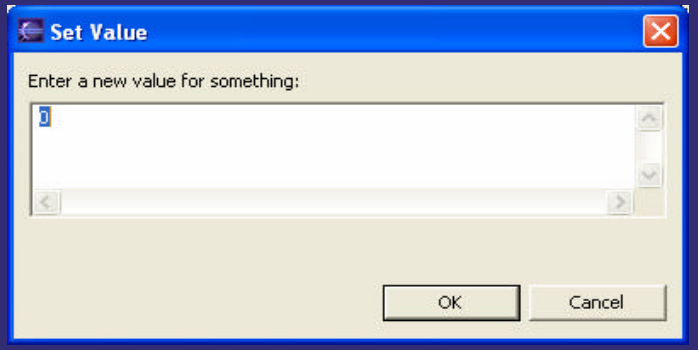
# Variables View

- Shows all fields of instance where breakpoint occurred
  - Select this to see all fields
  - Select any field to see value
  - If field is bound to an object, you can select Inspect from the menu to view its fields and values



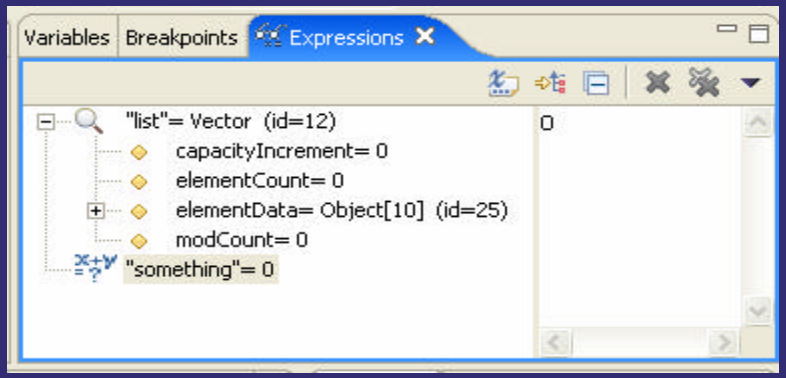
# Changing Field Values

- To change field value:
  - Select field in Variables view
  - Select Change Value… from the context menu
  - Enter new value into Set Variable Value window
  - Click OK



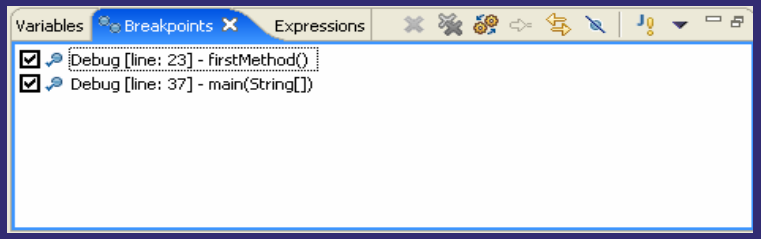
# Expressions View

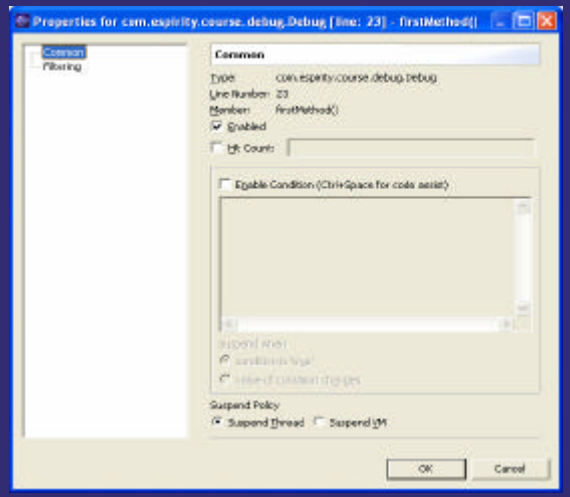
- Remembers all objects you have inspected
- Displays the fields of the object
  - You can see the values of the fields
  - You can Inspect the fields
- Opens when:
  - You Inspect an object
  - You click on the Watch from the context menu



# Breakpoint View

- Lists all available  breakpoints
- Can be used for manipulating breakpoints (through the views menu):
  - Enabling
  - Disabling
  - Removing
- Also displays breakpoints properties
- Accessed like other debugging views



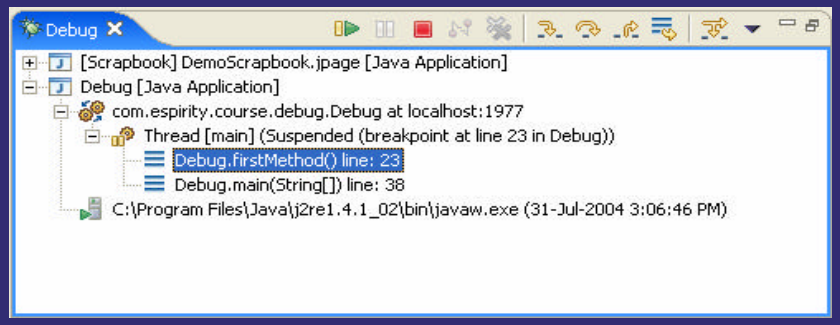



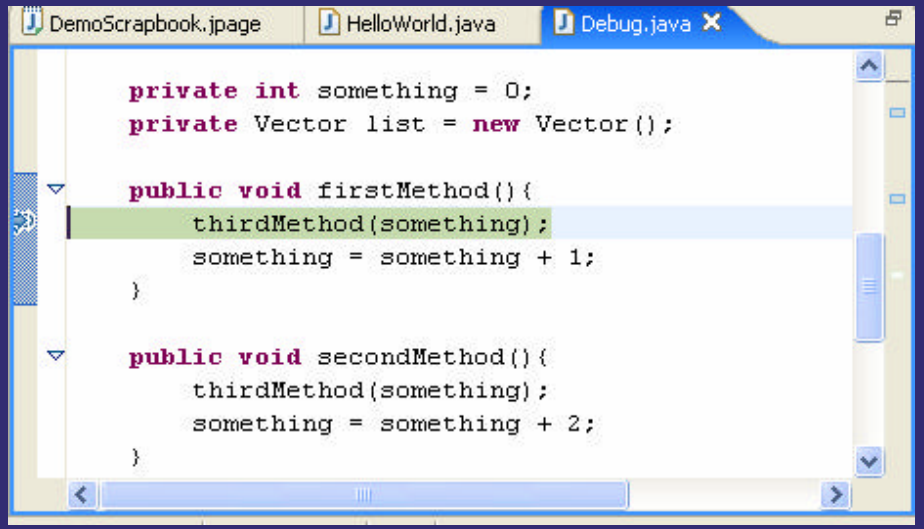


# Debug View

- Shows:
  - Active threads
  - Current stack frame when execution has stopped
  - Previous stack frames
- Method and variables are shown in the editor for the selected frame
  - Update in the editor updates the source





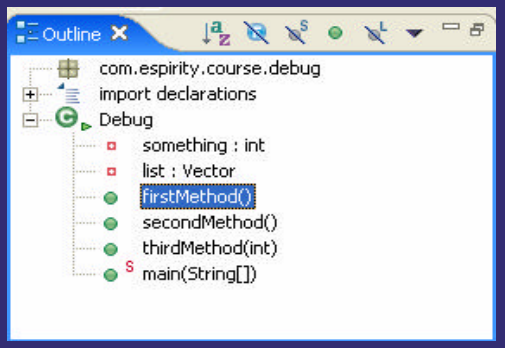
# Breakpoint Types

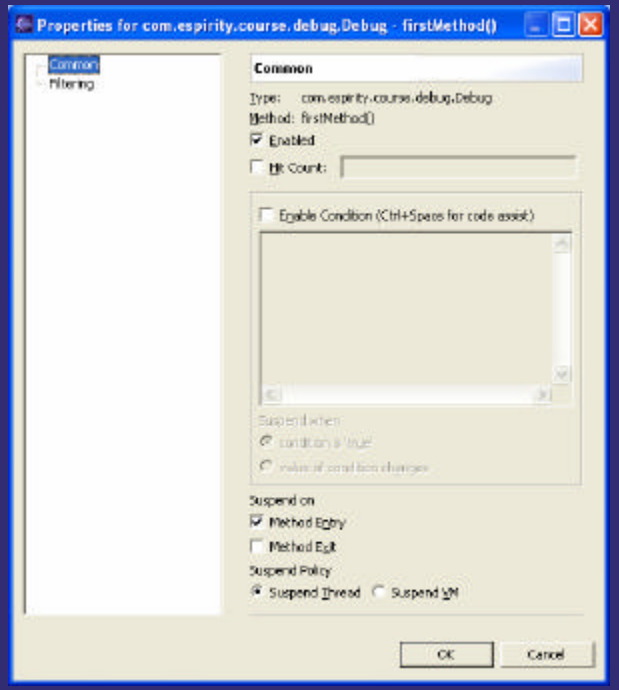
- Breakpoints can be set for the following Java entities:
  - Line  (simple breakpoint)
  - Method
  - Field (Watchpoint)
  - Java Exception
- Each breakpoint is set a different way and has different properties

# Method Breakpoints

- To set method breakpoint:
  - Select method in the Outline View
  - From context menu select Toggle Method Breakpoint
- To set breakpoint's properties:
  - Select breakpoint in editor
  - Select Breakpoint Properties.. from context menu
  - Set properties as desired
    - Entry, exit, enable hit count
- Execution suspends on entry/exit into method



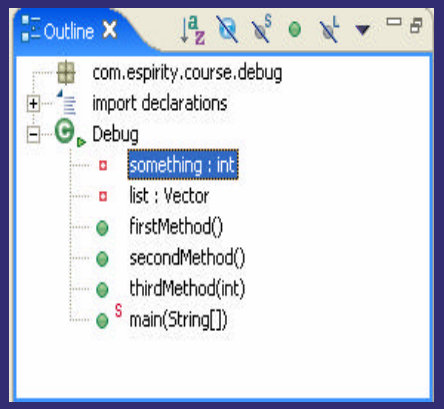



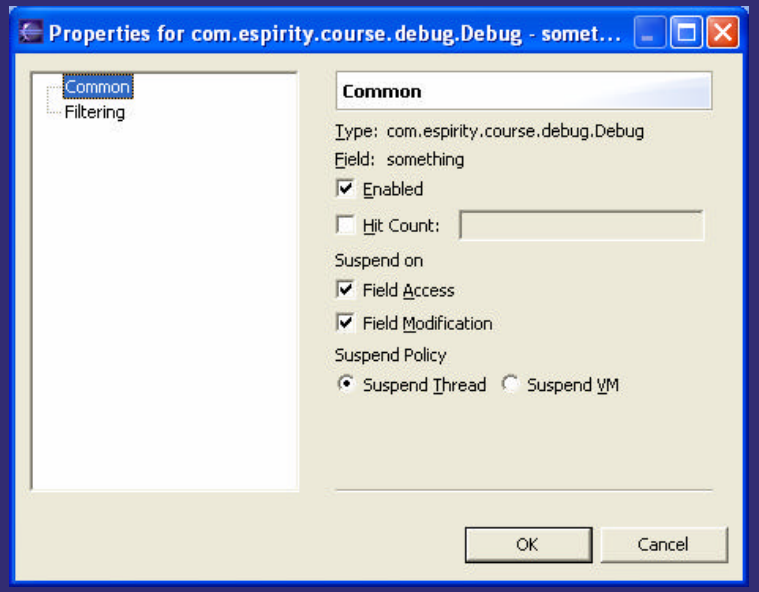


# Field Breakpoints

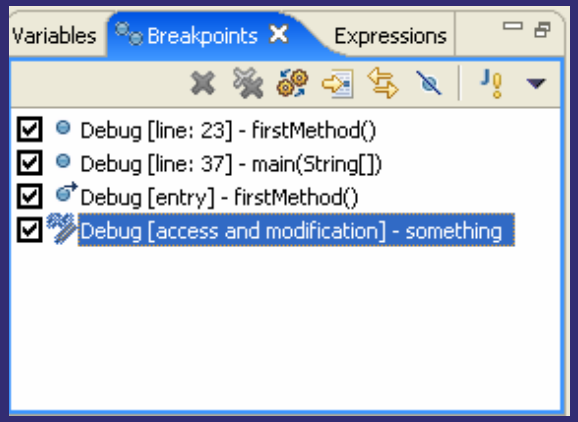
- Also known as watchpoint
- To set the watchpoint:
  - Select field in the Outline View
  - From context menu select Toggle Watchpoint
- To set watchpoint's properties:
  - Select breakpoint in editor
  - Select Breakpoint Properties.. from context menu
  - Set properties as desired
    - Access/modification, enable
- Execution suspended on access/modification of field

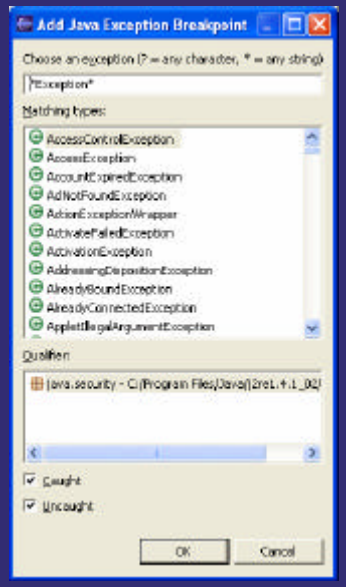


# Java Exception Breakpoint

- To Add Java Exception Point:
  - Select Add Java Exception Point from menu
  - Enter exception type
  - Specify what triggers a breakpoint:
    - Caught exception
    - Uncaught exception
    - Both





# How To Debug

- Here are simple steps for debugging in Eclipse:
  - Set your breakpoints
  - Hit a breakpoint during execution
  - Walk/step through code to other breakpoints
  - Follow along in editor
  - Inspect/Watch interesting fields
  - Watch the Console for things to happen

# Summary

- You have learned:
  - The views in the Debug Perspective
  - Typical debug session
  - How to use the Inspector
  - About the different types of breakpoints
  - How to set breakpoints
  - How step around your code doing debugging

# Labs!

Lab: Debugging in Eclipse