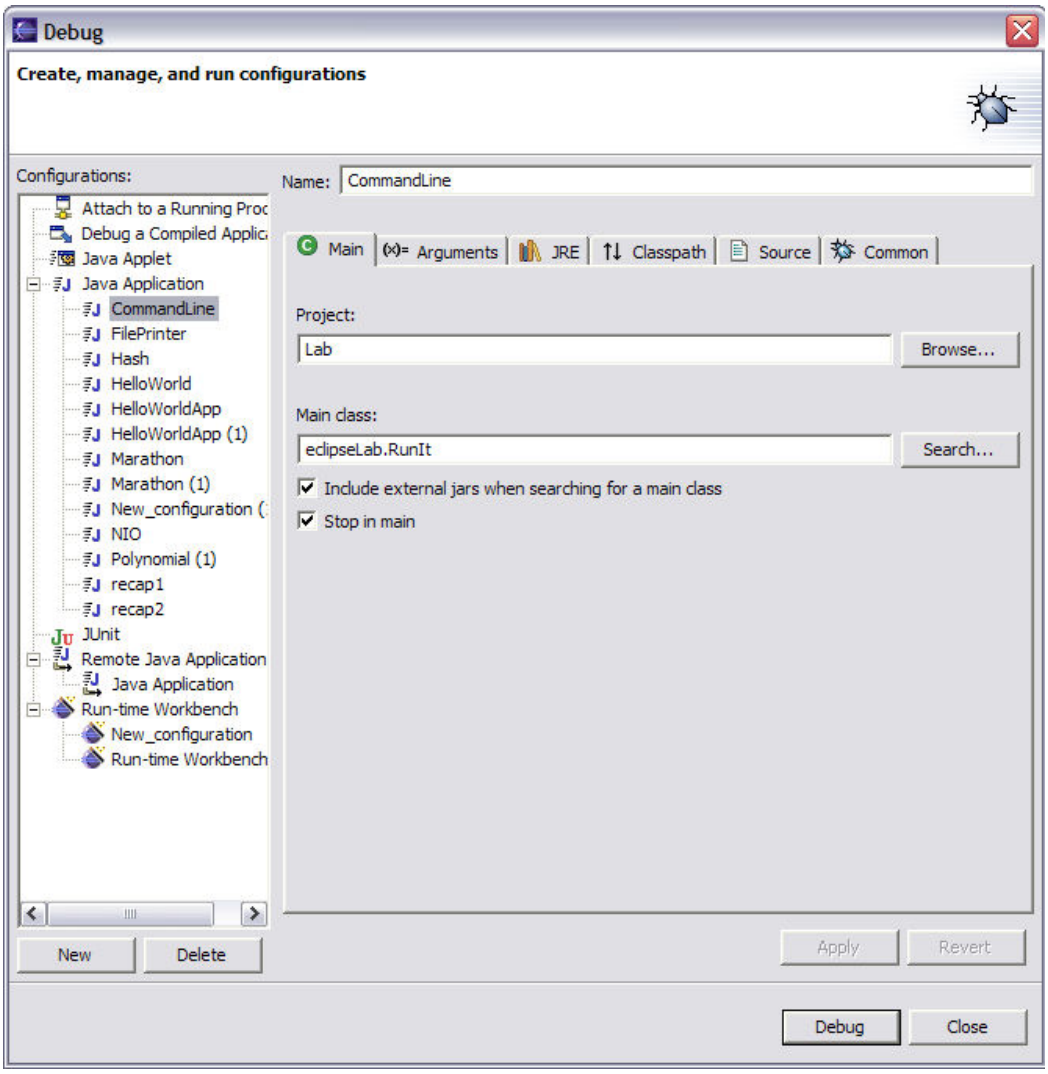
Java Development with Eclipse
Lab 2 The debug Perspective

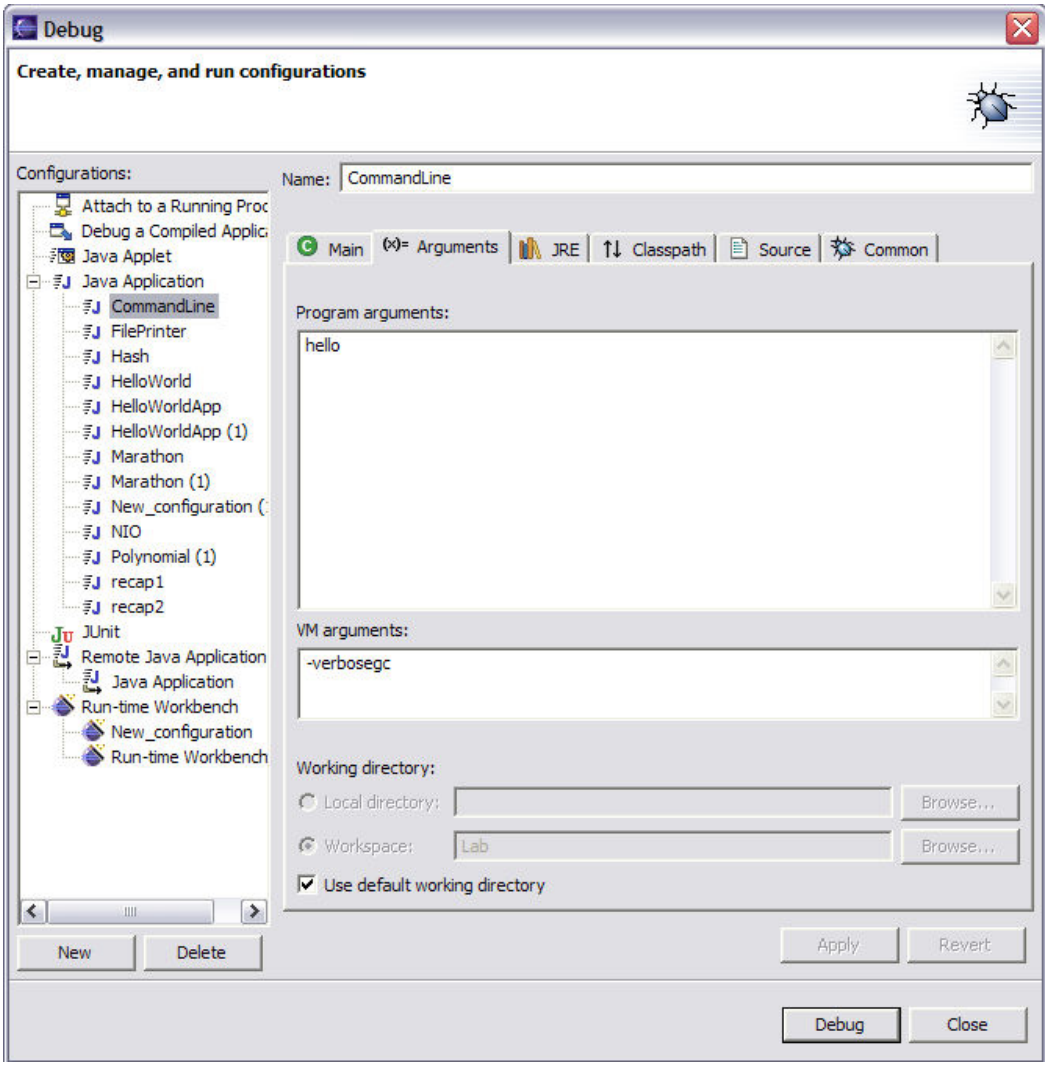
1. Debugging functionality

1. In the Resource perspective open the RunIt class in the java window.
2. Enter Run>Debug and the following window is displayed :



This is the debug configuration

3. 'Click' on the Argument tab and enter 'hello' in the 'Program Arguments' panel. See below.

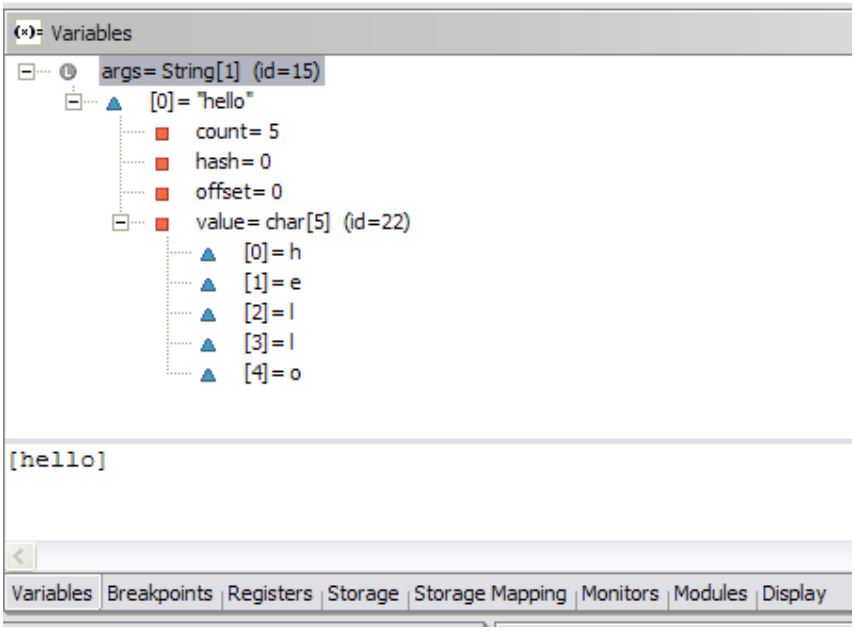


4. 'Click' on the 'Debug' button.
   This opens the 'Debug' Perspective shown in the current slide in the presentation.
   The RunIt application is being executed and is suspended at the first instruction.
   The blue arrow in the margin and the greyed line indicate where the programs is suspended.

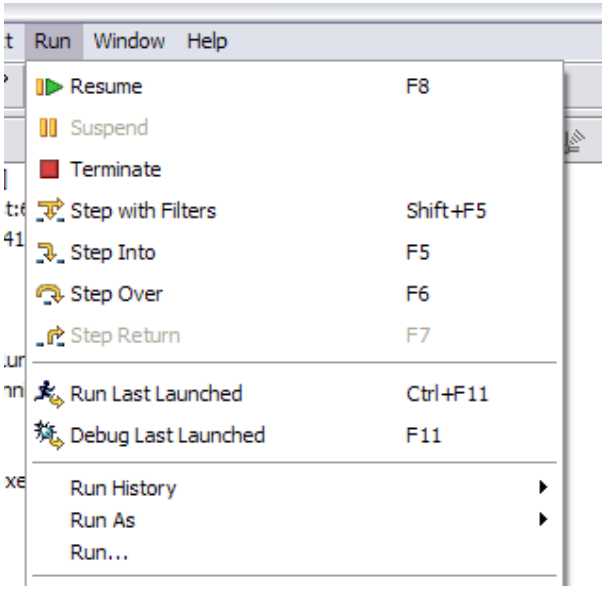
```
 * To change the template for this generated type
 * Window&gt;Preferences&gt;Java&gt;Code Generatio
 */
public class RunIt {

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

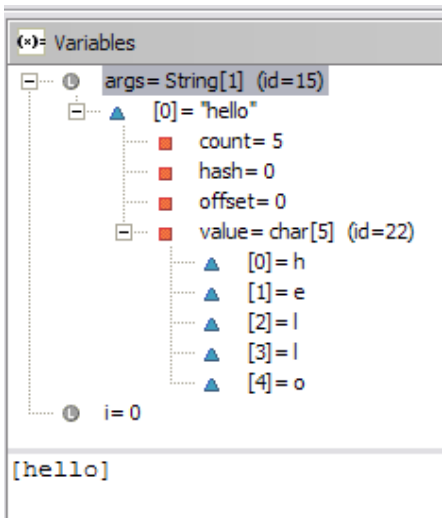
5. Look at the other windows in the debug perspective
   In the threads window the RunIt thread is suspended at line 20
   In the attributes window's variables tab the only known 'variable' the program argument 'hello' is displayed.
   Try 'opening' the 'args' variable and see the various components of a String array exposed.



6. Click on the run menu option and see the debugging functionality now enabled:
   Resume, terminate, step with filters, step into …



7. ‘Click’ on ‘Step Over’. The variable ‘i’ now appears in the variable tab:

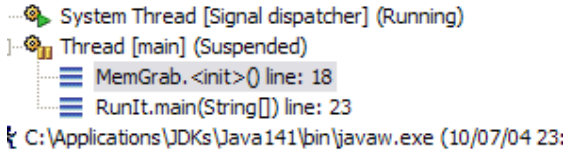


8. ‘Click’ on ‘Step Over’ or press F6 until ‘i’ is 10.
   See how the program is executing 1 line at a time.

9. Step through the program until the execution is suspended at line

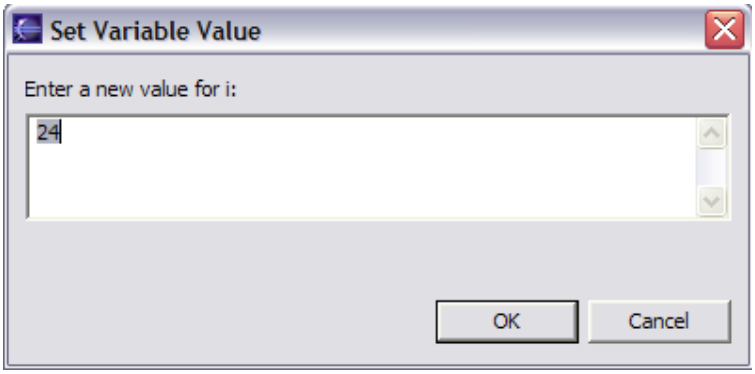
```
MemGrab aGrab = new MemGrab();
```

   Now enter F5 or ‘Step Into’ and see the program suspend in the MemGrab() constructor. In the thread window the java stack can be seen. RunIt.main(String[]) method calls MemGrab.<init>() method,



10. Now enter F7 or ‘Step Return’ to return to the RunIt.main method.

11. Place the cursor over ‘i’ in the variable tab and ‘double click’ to relieve the following window. ‘i’ can now be changed. Change it to ‘82’ and click on OK.



12. Step through the program until the execution is suspended at line

    MemGrab aGrab = **new** MemGrab();

    Now depress the Ctrl + Shift + b keys. A breakpoint has been set at this source code line. Now enter F8 or 'Resume', program execution now proceeds to the next breakpoint. Enter F8 a few times and each time see 'i' incremented to indicate that a whole loop Iteration has been executed.

13. Click on the 'breakpoints' tab to see the breakpoint added in section 12.

14. Try 'hot code replacement'. Suspend the java application on a breakpoint. Modify the code, for example, change RunIt.java with the following 1 line addition:

```
int i=0;
System.out.println("Lab1");
while (i++ < 1000) {
      MemGrab aGrab = new MemGrab();
      i++;
}
```

    Enter 'Ctrl s' to save RunIt.java. If you have automatic compilation enabled RunIt.java will be recompiled. Continue editing the application. You have changed the application code during an execution. Stepping through notice how I is incremented twice in each loop body as a result of this code modification.

15 To end the lab select either the terminate debug menu option or continue stepping though the code.