



DEBUGGING

KAUSTAV DAS SHARMA

WHAT IS DEBUGGING?

- *Debugging is the routine process of locating and removing bugs, errors or abnormalities from programs.*



WHY DEBUG?

- Three types of errors

- Compilation Error:

- Mostly syntactical errors which are shown by the IDE or diagnosed upon attempt at compilation hence a debugger is unnecessary.

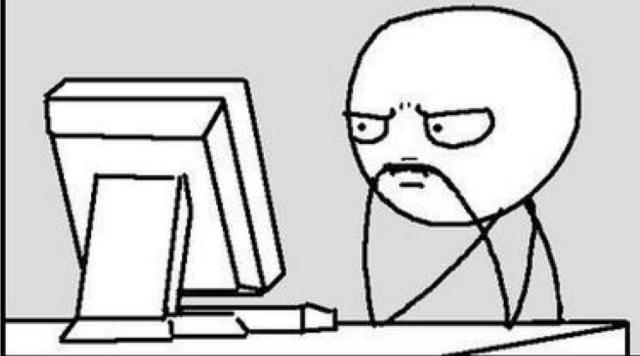
- Run-time Error:

- Exceptions, I/O issues. Easy to trace type but potentially difficult to diagnose the cause, hence debugging is helpful.

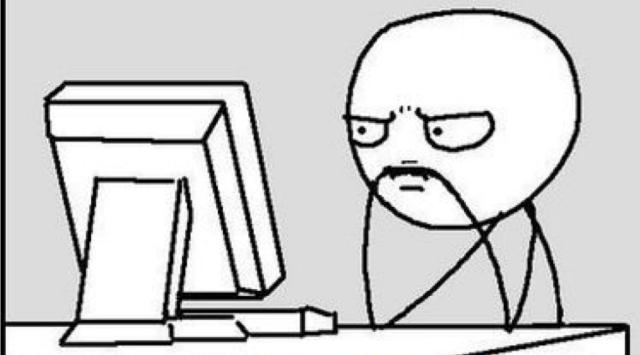
- Logical Error:

- Incorrect output. Hard to trace type and difficult to diagnose the cause, hence debugging is necessary.

It doesn't work..... why?



It works..... why?



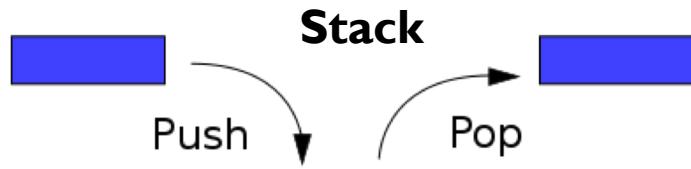
MemeBlender.com <<<<<<<<<<<<

HOW DOES A DEBUGGER WORK?

- **Basic utility of the debugger includes:**
 - Executing a process
 - Single stepping through a process
 - Breakpoints
 - Variable values
 - Stack traces
- **Memory Management in Java:**
 - Application allocated a certain amount of RAM (Random Access Memory) and the Java Virtual Machine organizes this memory.

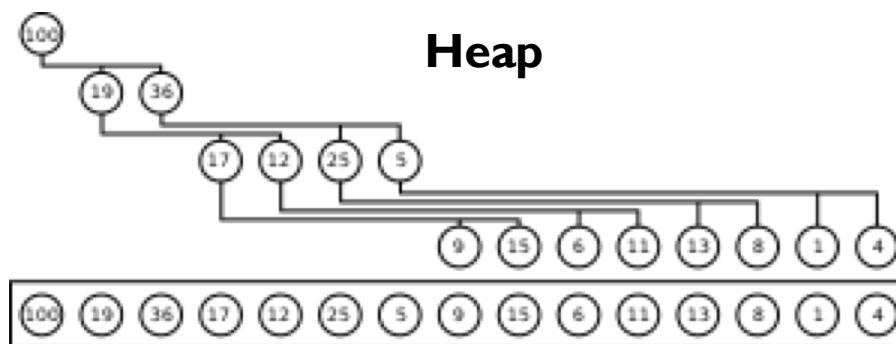
Stack memory:

- Stack memory stores variable values when their methods are called and stores references to objects which are located in heap memory. A “block” of memory is erased when method ends, and a new “block” is created.



Heap memory:

- Heap memory stores objects and classes, and is always available in the heap space therefore having global access.

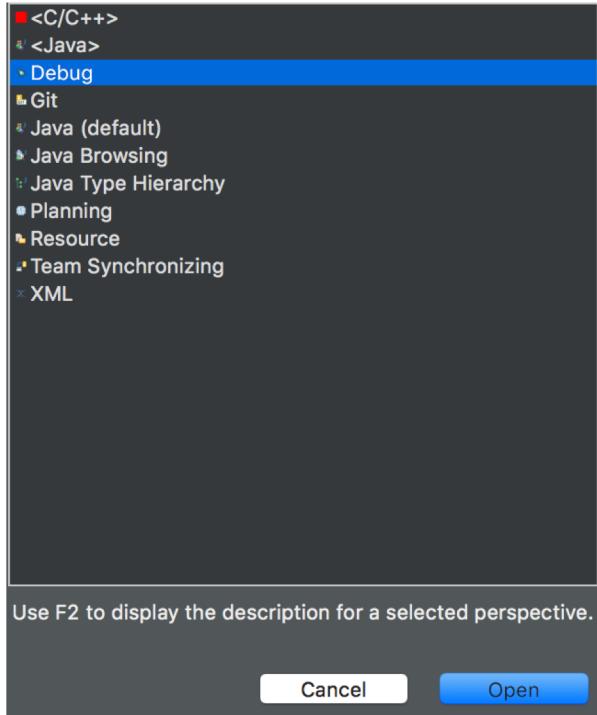


Garbage Collection:

- Clears any objects which aren't referenced in methods and other advanced memory management techniques.

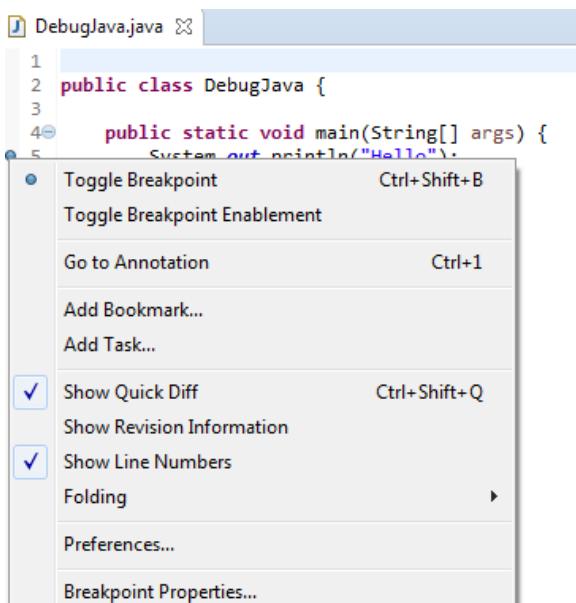
DEBUGGING IN ECLIPSE

- To enter Debugger view:



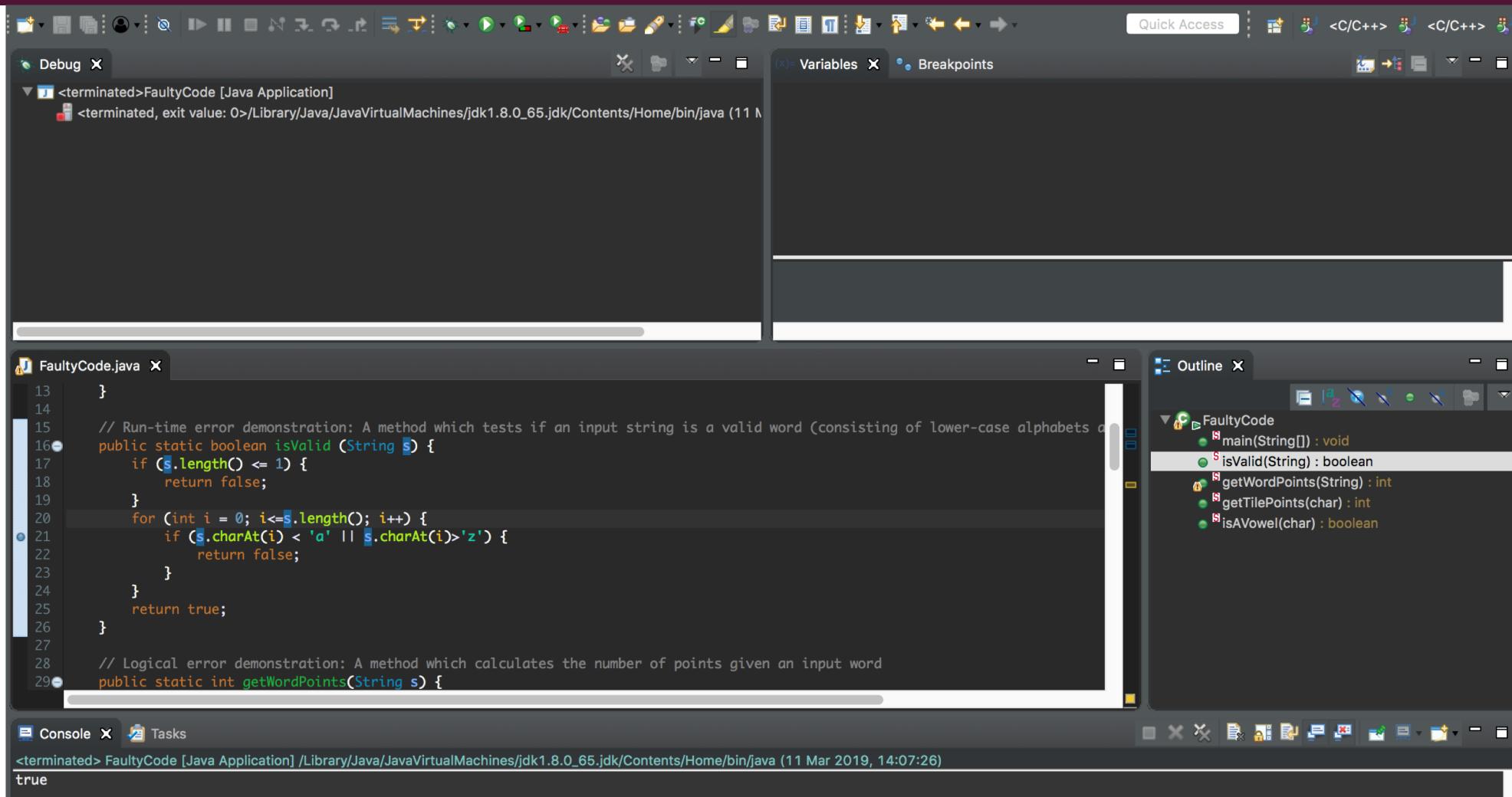
■ Breakpoints:

To define a breakpoint in your source code, right-click in the left margin in the Java editor and select Toggle Breakpoint. Alternatively, you can double-click on this position.



- **Contents of Debug perspective:**
- **Debug view** – Visualizes call stack and provides operations on that.
- **Breakpoints view** – Shows all the breakpoints.
- **Variables/Expression view** – Shows the declared variables and their values. Press Ctrl+Shift+d or Ctrl+Shift+i on a selected variable or expression to show its value. You can also add a permanent watch on an expression/variable that will then be shown in the
- **Expressions view** when debugging is on.
- **Display view** – Allows to Inspect the value of a variable, expression or selected text during debugging.
- **Console view** – Program output is shown here.

DEBUGGING IN ECLIPSE



Shortcut	Toolbar	Description
F5 (Step Into)		Steps into the call
F6 (Step Over)		Steps over the call
F7 (Step Return)		Steps out to the caller
F8 (Resume)		Resumes the execution
Ctrl + R (Run to Line)		Run to the line number of the current caret position
Drop to Frame		Rerun a part of your program
Shift + F5 (Use Step Filters)		Skipping the packages for Step into

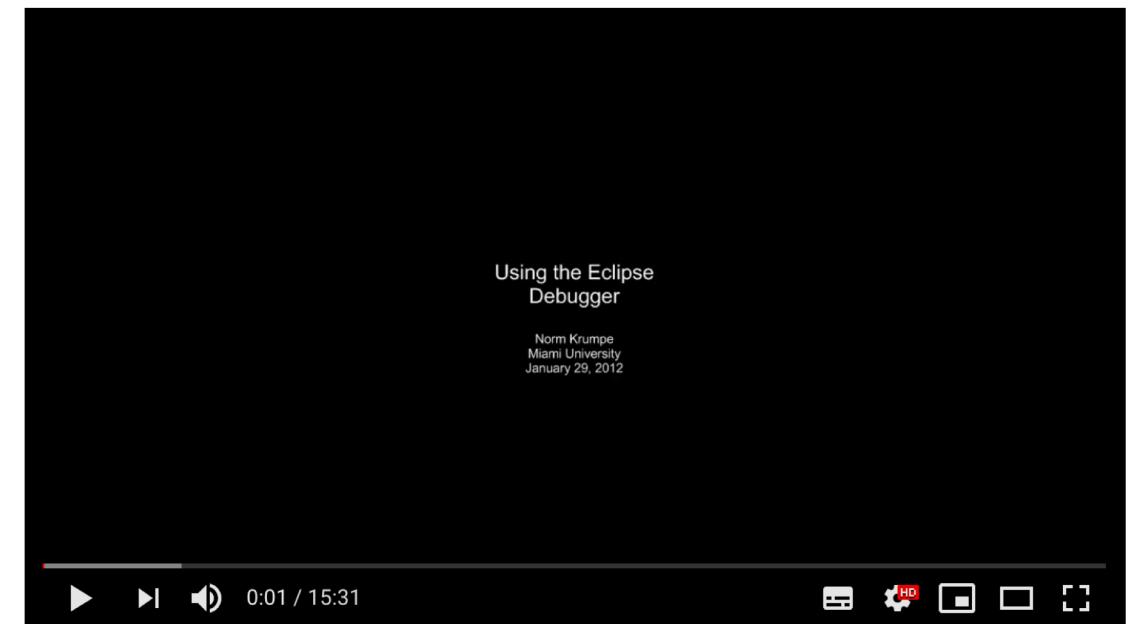
DEBUGGING IN ECLIPSE

RESOURCES FOR FURTHER INVESTIGATION

The screenshot shows a web page from the Eclipse Foundation. At the top, there's a dark header with the Eclipse Foundation logo on the left and 'Members' and 'Working Groups' links on the right. Below the header, a breadcrumb navigation shows 'Home / Eclipse Newsletter / 2017 / Eclipse Oxygen - A Breath of Fresh Air / Debugging the Eclipse IDE for Java ...'. The main content area has a dark background with a geometric pattern. The title 'Debugging the Eclipse IDE for Java Developers' is centered in white text. Below the title, a paragraph of text explains what debugging is and how the Eclipse Java IDE provides tools for it.

Debugging is the routine process of locating and removing bugs, errors or abnormalities from programs. It's a must have skill for any Java developer because it helps to find subtle bug that are not visible during code reviews or that only happens when a specific condition occurs. The Eclipse Java IDE provides many debugging tools and views grouped in the **Debug Perspective** to help the you as a developer debug effectively and efficiently.

https://www.eclipse.org/community/eclipse_newsletter/2017/june/article1.php



Using the Eclipse Debugger

https://www.youtube.com/watch?v=9gAjlQc4bPU&ab_channel=NormKrumpe