

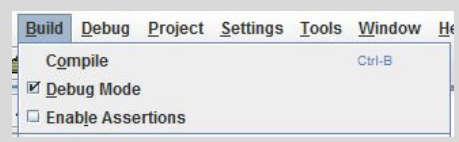








Debugging in jGRASP (CIST1400)

Inspired by http://jgrasp.org/tutorials187/06_Debugger.pdf



Why do I need a debugging tool?

jGRASP Debugging Commands

Icon / Command	Description	Comments
	Confirm that the Debug Mode is checked in the build options from the taskbar.	After Debug Mode is checked the program can be compiled to be debugged.
	Right click upon a line in your code and select this option to insert a breakpoint.	The debugger will pause at breakpoints.
	Open debugger canvas in new window	Variables from the original debug tab can be dragged to this window.
	Step forward in program	Move forward to next breakpoint.
	Step to cursor in program	Move forward until cursor. Skips breakpoints.
	Auto step	Automatically step after hitting step once.
	Pause debugger	Pauses program at current breakpoint.
	Resume debugger	Similar to step. Resumes paused program to next breakpoint.
	Auto resume	Similar to auto step but only after hitting resume once.

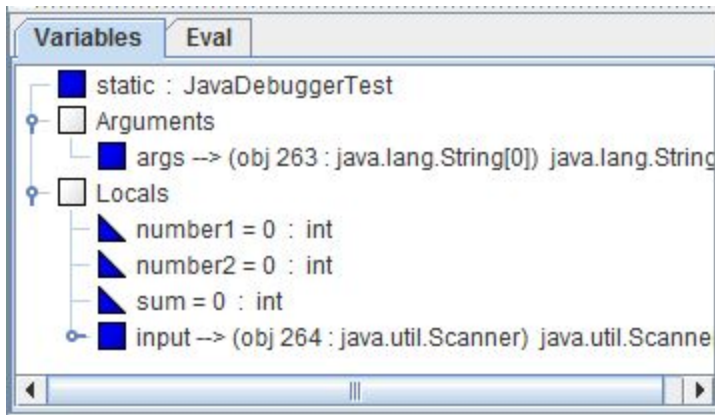
Debugging in jGRASP (CIST1400)

Inspired by http://jgrasp.org/tutorials187/06_Debugger.pdf



Viewing Variables in Memory

Upon running a program in debugging mode, a debug tab will appear on the left side of the screen. This tab will have a window labeled as “Variables.” This window will display all initialized variables and their current values as the program is stepped through breakpoints. Your created variables will be listed under the “Locals” category. If working inside a method, only variables declared in that particular method will be currently listed.



Watching for Access

General Debugging Strategies

- Put a breakpoint before and after where you think the problem is.
- Have a breakpoint any time you change the value of a variable.
- Put a breakpoint before and after every loop.
- Put a breakpoint inside a loop to view every iteration.