Eclipse IDE Cheat Sheet

Beginner's guide on navigating through Eclipse

By Alizah Saqib and Deb Harding

Contents

IDE Overview and Eclipse

What is an IDE?
An Integrated Development Environment (IDE) is a software application that helps you write, edit, and manage code — all in one place.

Think of it as a toolbox for coding that combines:

- A text editor (to write your code)
- A preview or terminal window (to test or run your code)
- Helpful tools like:
  - Syntax highlighting (color-coded tags and text)
  - Autocomplete (suggests code as you type)
  - Error checking (shows when something is wrong)
  - Debugging (Step through code to understand what is happening or find errors

Eclipse IDE

Different IDEs can be used for different languages and use cases. For this class I will use Eclipse.

Why and IDE?

- Makes coding easier and faster
- Helps you avoid mistakes
- Organizes your project files in one place
- Gives you immediate feedback while you work

Why Eclipse?

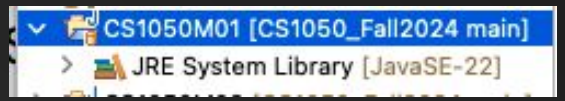
- Free & Reliable — No cost, works across Windows, macOS, and Linux.
- Industry-Standard Tool — Still widely used in Java development.
- Java-Focused Features — Offers auto-completion, auto-formatting, debugging, and more to make coding smoother.
- Real-World Learning — Helps you practice organization, testing, and debugging in a professional environment.

[10 Best Java IDE For Developers in 2025 - GeeksforGeeks]

Create Project

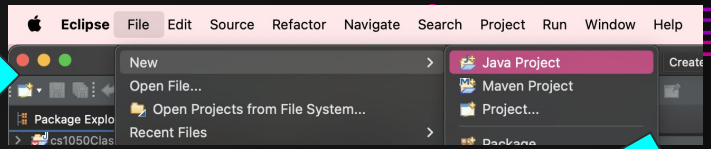
1. Start a new project click file > new > java project
2. Enter a name for your project (e.g., cs1050M01)
3. Configure settings -
   a. Checkmark “Use default location” which should be your workspace folder in your git repository
   b. “Use an execution environment JRE” should have your JavaSE (e.g. “JavaSE-22”)
   c. Project layout: checkmark “use project folder as root for sources and class files”
   d. Module: checkmark “Create-module-info.java file” and “Generate comments”
4. Click “finish” to create the project
5. Check to make sure the project is in the explorer window on the left and has JRE in it as shown

Step 5



This creates a project folder to organize your java files

Step 1



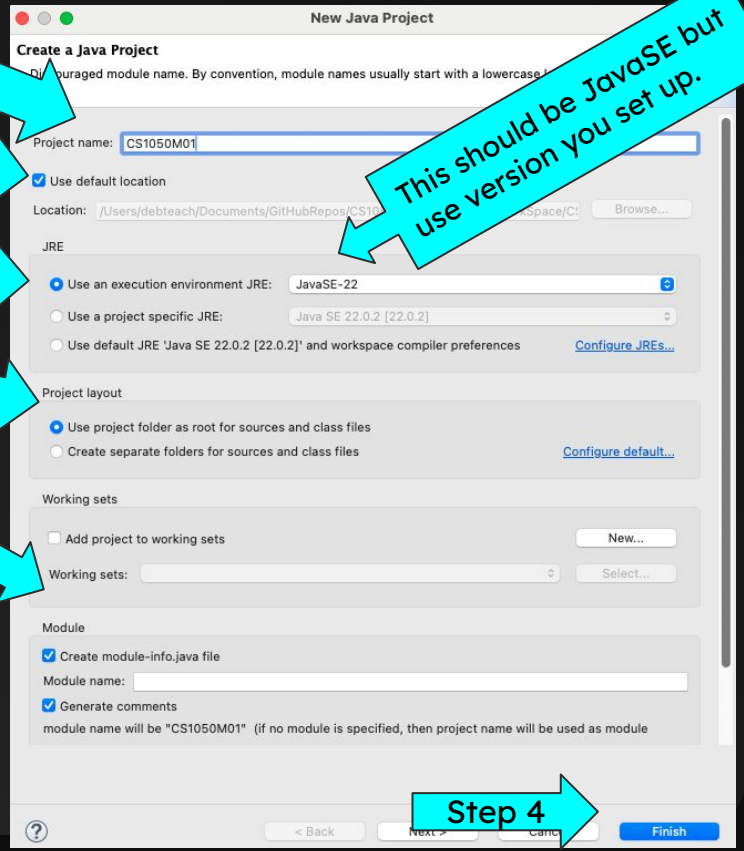
Step 2

Step 3a

Step 3b

Step 3c

Step 3d



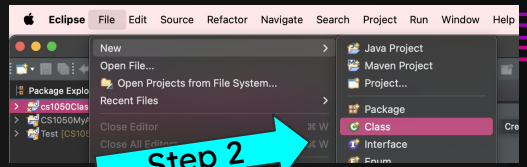
This should be JavaSE but use version you set up.

Step 4

Create Class & Java File

1. Select the project folder you want to create it in - in the package explorer on the left, click on your project to highlight it
2. Create a new class - click file > new > class
3. Configure settings -
   a. Source folder should be set, if not, select the project folder you want it to go in
   b. Package should be empty, if not, delete the package
   c. Enter a name for your class (e.g., GEM01Calculation)
   d. Modifiers: checkmark “public” and “package”
   e. Method stubs: checkmark “public static void main(String[] args)
   f. Comments: checkmark “generate comments”
4. Click finish to create the class
5. This create your java file with class and main method

Step 1



Step 2

Step 3a

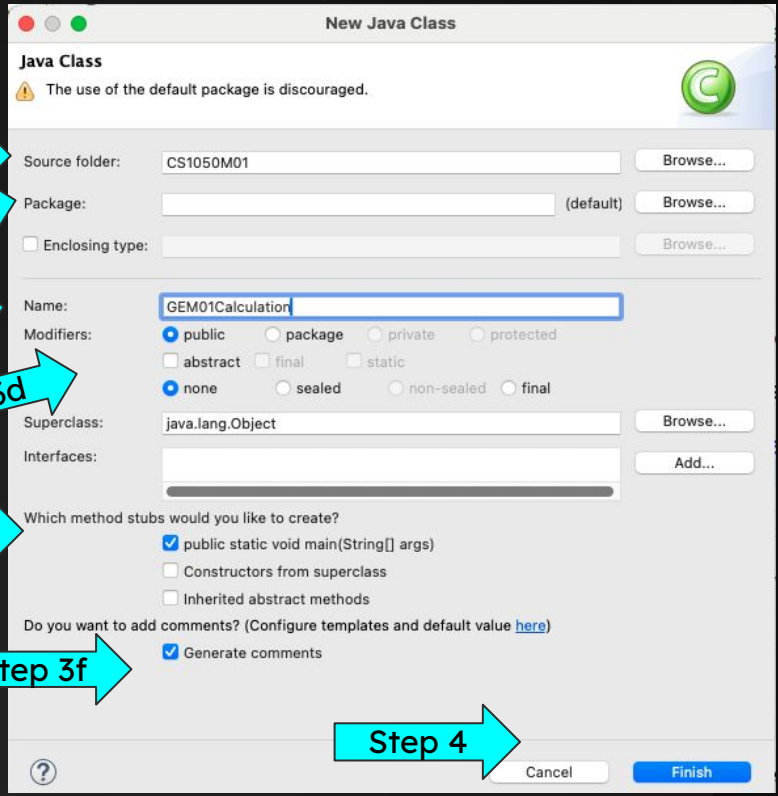
Step 3b

Step 3c

Step 3d

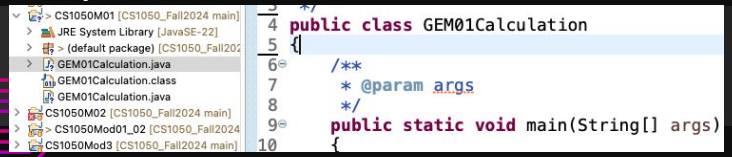
Step 3e

Step 3f



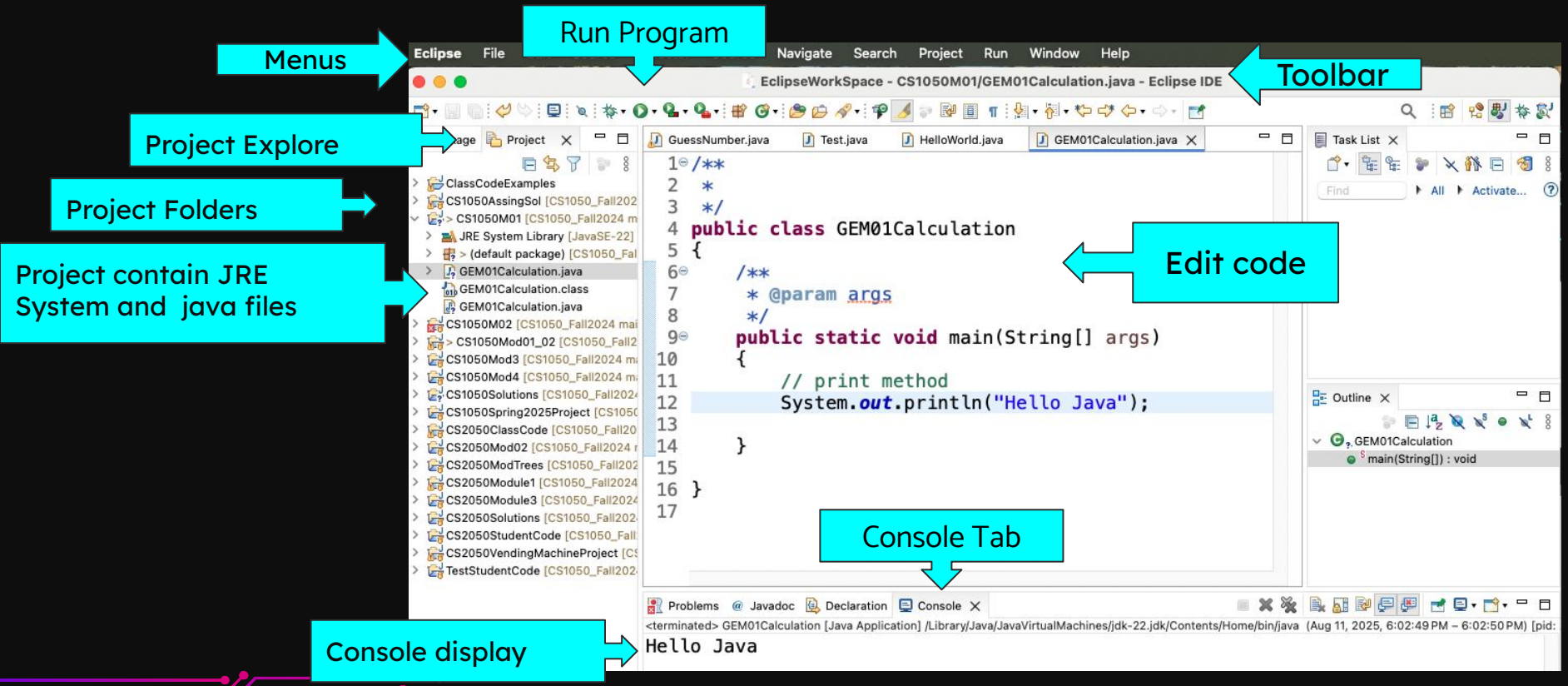
Step 4

Step 5



Navigating Java Perspective

Running your code



Edit, Run and Compile

When you click Run (▶) in Eclipse, two things happen:

1. Save & Compile – Eclipse automatically saves your changes and compiles your .java source files into .class files (Java bytecode).
2. Run on the JVM – The compiled .class files are then executed by the Java Virtual Machine (JVM).

Compilation checks your code for syntax errors and turns it into a format the JVM can understand. If there are errors, Eclipse won't run the program until they're fixed.

Warnings don't stop compilation, but they should still be fixed to avoid future problems.

Editing Code:

- You can change your code anytime in the editor.
- When you run the program again, Eclipse automatically recompiles the changed files.
- You don't have to manually compile in most cases—Eclipse handles it for you.

Errors

Error

Error Location - hover over for information on how to fix.

Problem

Problems Tab

Shows errors from all files.

Error Information

Description

File name

Line Number

Console Tab

Program won't run and displays error in Console window

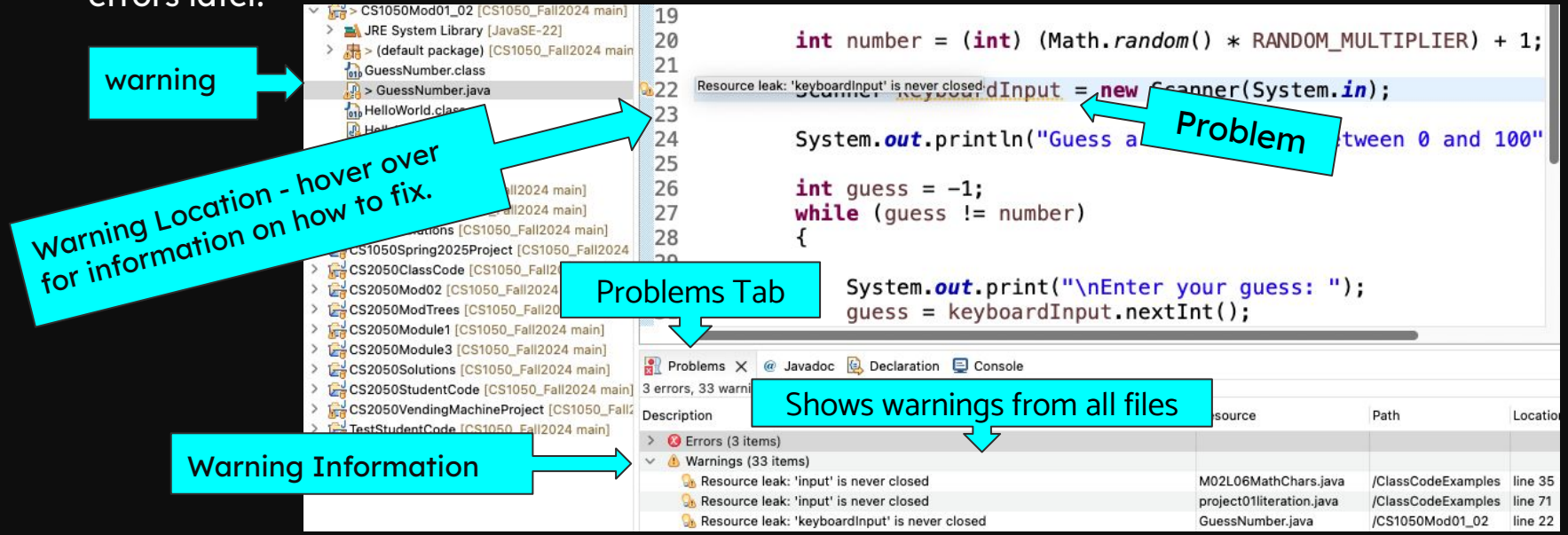
```
Problems  @ Javadoc  Declaration  Console
<terminated> GuessNumber (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk-22.jdk/Contents/Home/bin/java (Aug 11, 2025, 6:09:22 PM – 6:09:22
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        Syntax error, insert ";" to complete BlockStatements

        at GuessNumber.main(GuessNumber.java:18)
```

Warnings

Your program should not contain warnings. Why fix them?

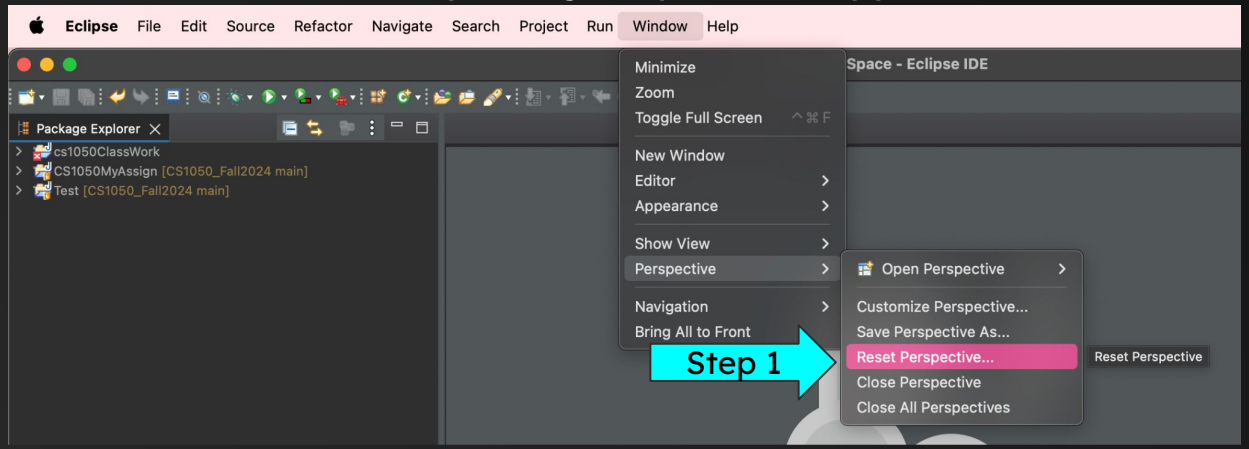
- Possible Security Risks – Some warnings indicate unsafe code that could be exploited.
- Maintainability – Clean code is easier for you (and others) to read, debug, and improve.
- Professional Standards – In industry, code reviews often require zero warnings before approval.
- Prevent Future Bugs – Warnings can signal logic issues or outdated methods that might cause errors later.



Perspective and Views

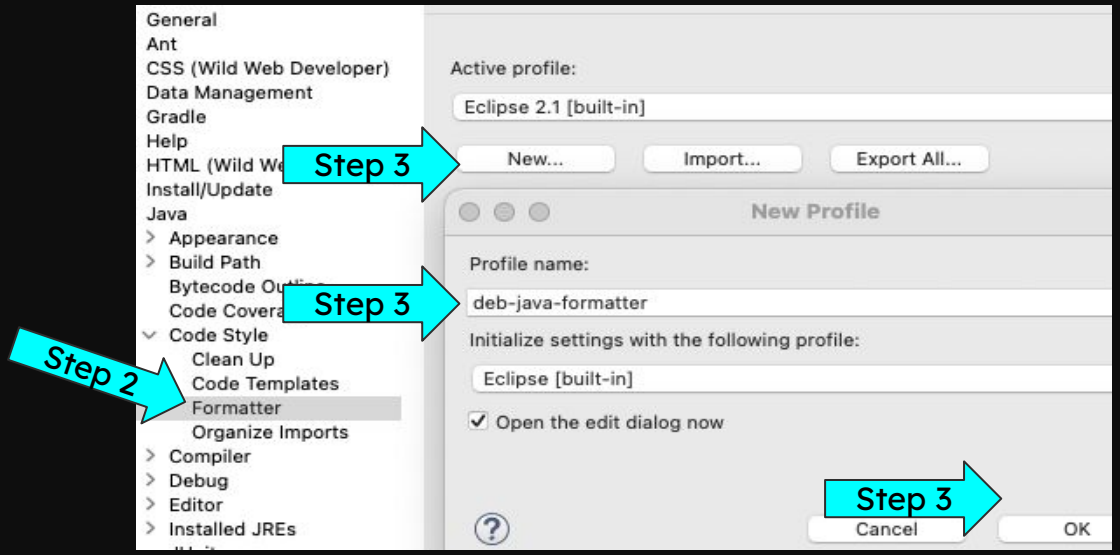
A “perspective” is just a layout of views and controls what you see in certain menus and toolbar. You will be using the Java and Debug Perspectives

1. If your “perspective” gets messed up Click window > perspective > reset perspective
   a. If you accidentally close a panel and want to bring it back
   b. If important windows like the console or package explorer disappear



Formatter Profile Set Up

1. Go to settings (or might be called preferences)
   a. Windows OS Go to menus Window -> preferences
   b. Mac OS go to menus Eclipse -> settings
2. Then navigate to Java -> Code Style -> Format
3. Click new, Give your profile a name and click ok
4. Go to the Braces section, for all categories set Position to Next line.
5. Click ok to save profile.
6. Click apply and save

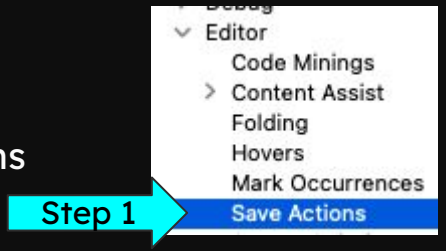


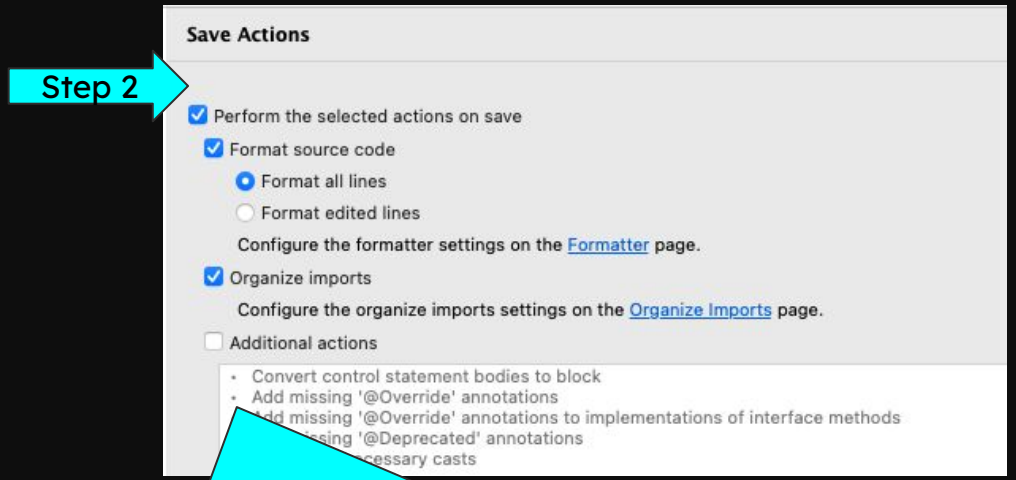
This helps find error when brace is missing or incorrectly placed.

Auto-Format on Save

Save Actions on, running code will always auto-format since Eclipse saves before compiling.

1. Go to
   a. Mac: Eclipse → Settings → Java → Editor → Save Actions
   b. Windows: Window → Preferences → Java → Editor → Save Actions
2. Select these save actions





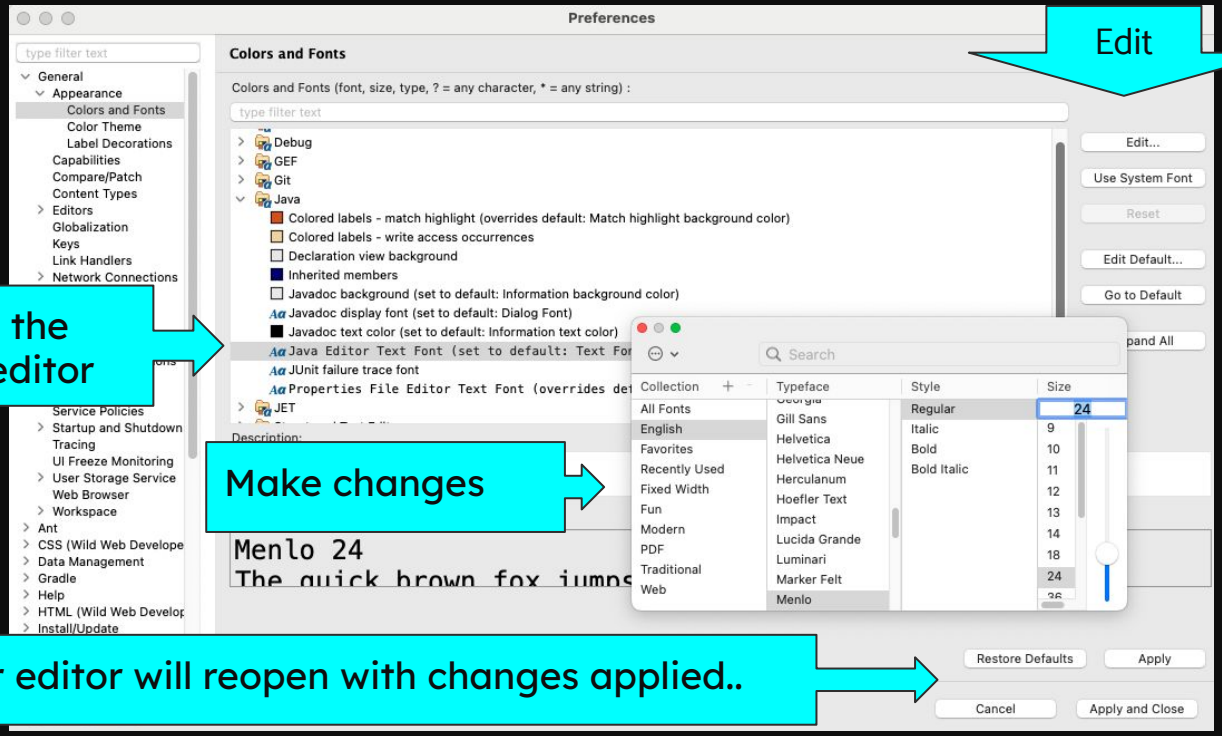
Type a messy block of code, then click Run — you should see the the correct format.

Change Themes, Colors Fonts

[6 Best Fonts for Coding to Keep Your Eyes from Eyestrain | by plabs.id]

If you want change the defaults go to Settings/Preferences general-> Appearance ->

- Colors and Fonts:
- Color Theme



For example, you can edit the font and size for the text editor

Make changes

Apply and your editor will reopen with changes applied..

Debug - Set Breakpoints

Set Breakpoints to step through each statement and see what is stored in each variables memory

a. To set a breakpoint, double click directly to the left of the line numbers
b. A blue circle will appear next to the line, indicating that a breakpoint has been set
c. To get rid of the breakpoint, double-click on it and the blue dot will disappear

Breakpoint

Breakpoint

EclipseWorkSpace - cs1050ClassWork/GuessNumber.java - Eclipse IDE

GuessNumber.java

```
1  /**
4
5  import java.util.Scanner;
6  /**
7   * This program takes user input to see if they can guess the
8   * random number generated
9   */
10 public class GuessNumber {
11
12     /**
13      * @param args
14      */
15     public static void main(String[] args) {
16         // Main method
17         // Line 18 generates a random number between 0 and 100
18         int number = (int)(Math.random() * 101);
19         // Line 20 declares and initializes the variable input
20         Scanner input = new Scanner(System.in);
21         System.out.println("Guess a magic number between 0 and 100");
22         // Line 23 declares and initializes variable guess with initial value as -1
23         int guess = -1;
24         while (guess != number) {
25           System.out.print("\nEnter your guess: ");
26           guess = input.nextInt();
27           // Depending on users input, next line will either show yes, too high, or too low
```

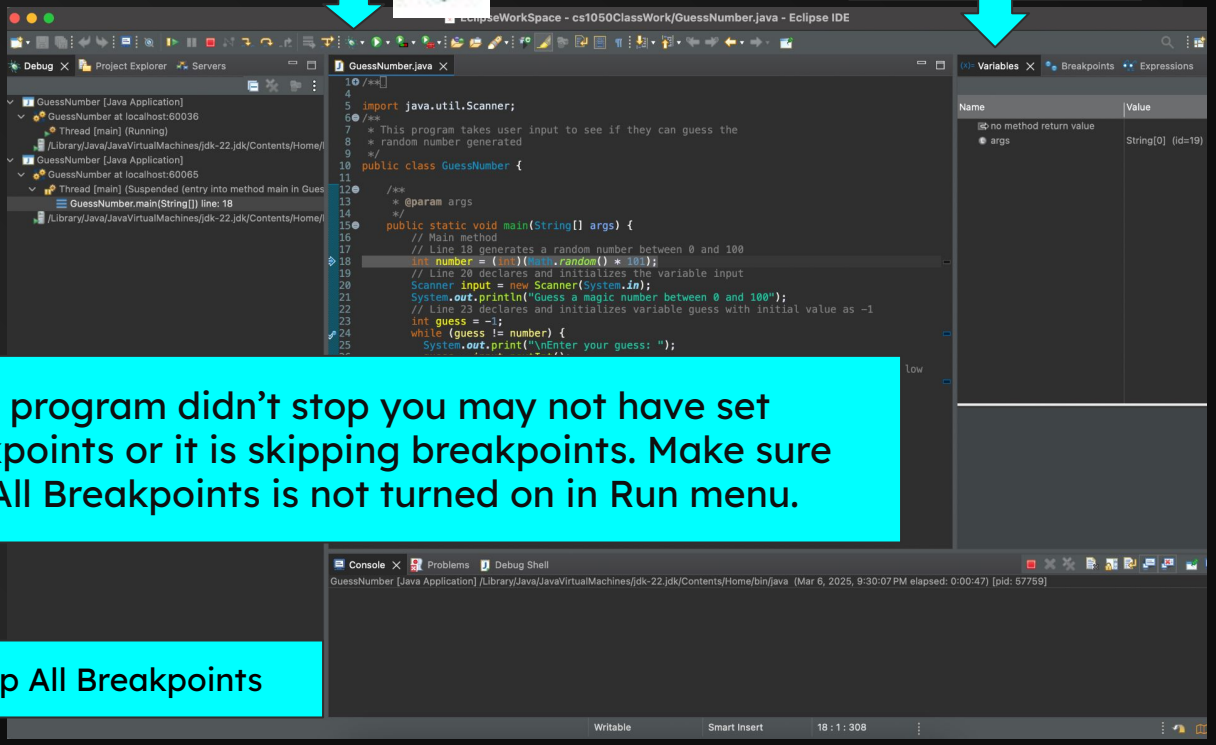
Debug: Start Debugging Session

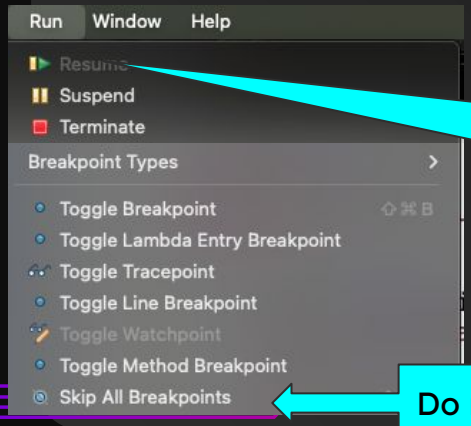
To start a debugging session click the debug icon (a green bug) in the toolbar before the run button. You should see the variables window (Debug Perspective). If you don't this could be because

- You didn't create your code in a project
- The java code has syntax errors

Variables window

Debug window





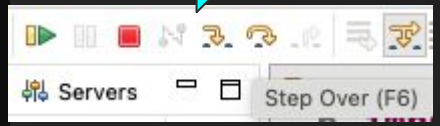
If the program didn't stop you may not have set breakpoints or it is skipping breakpoints. Make sure Skip All Breakpoints is not turned on in Run menu.

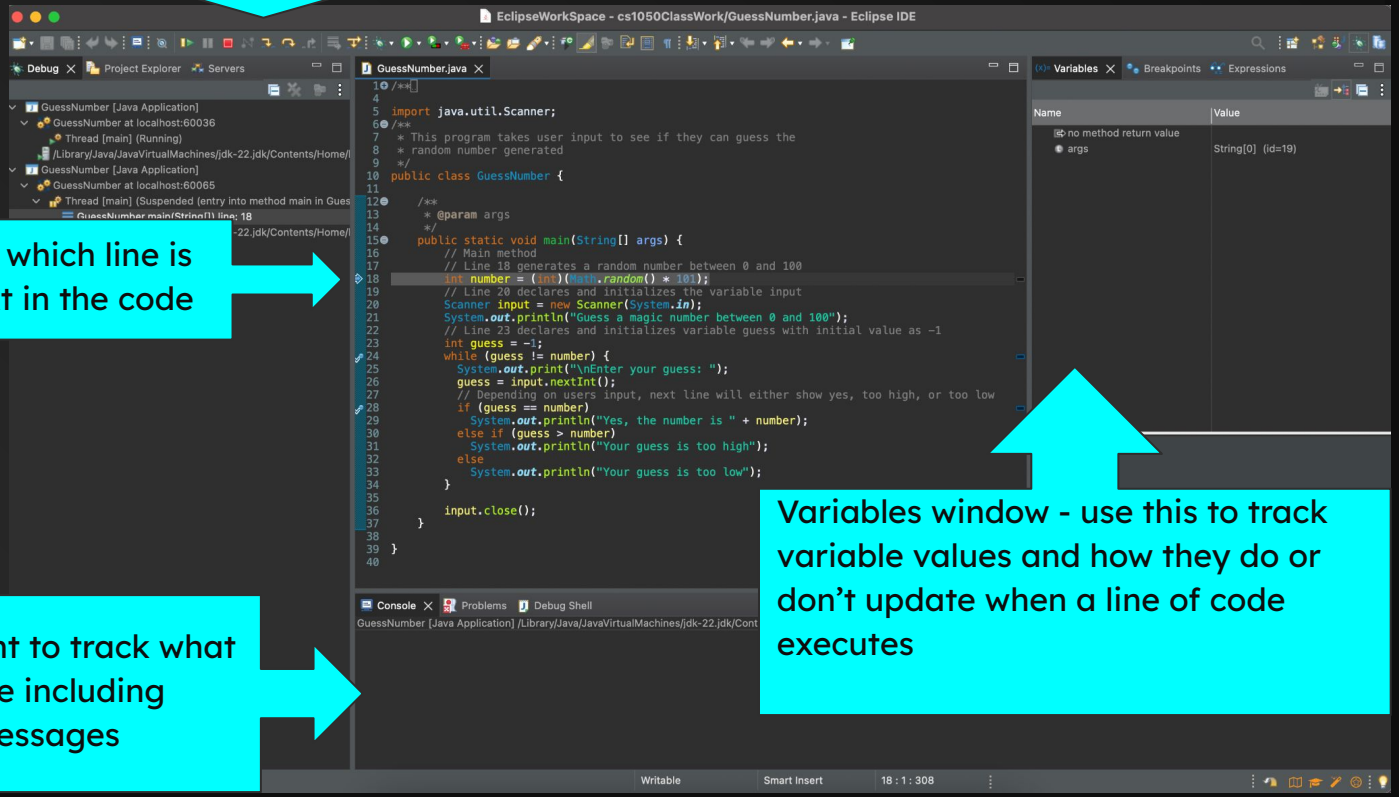
Do not select Skip All Breakpoints

Debug: Step Through Code

Use the “step over” button to control the flow of the program and move through each line manually.

Menus





The pointer tells us which line is to be executed next in the code

Variables window - use this to track variable values and how they do or don’t update when a line of code executes

Console output - it’s important to track what is being printed to the console including print statements and error messages

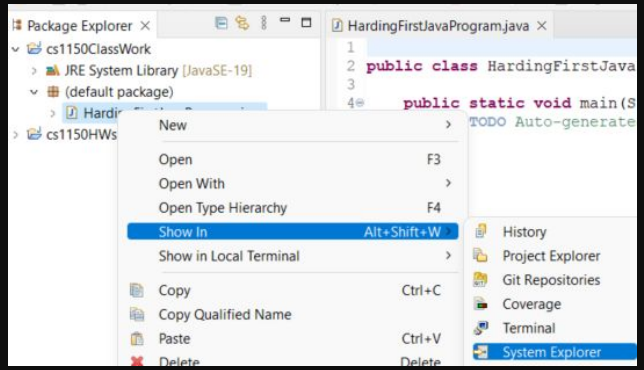
Eclipse Keyboard Shortcuts

Action	Mac	Windows
Run Program	⌘ + F11	Ctrl + F11
Debug Program	F11	F11
Format Code	⌘ + ⇧ + F	Ctrl + Shift + F
Organize Imports	⌘ + ⇧ + O	Ctrl + Shift + O
Search / Find	⌘ + F	Ctrl + F
Find in Project	⌘ + H	Ctrl + H
Open Type (Class)	⌘ + ⇧ + T	Ctrl + Shift + T
Open Resource (File)	⌘ + ⇧ + R	Ctrl + Shift + R
Rename Variable/Method/Class	Alt + ⌘ + R	Alt + Shift + R
Switch Tabs	⌘ + Option + → / ←	Ctrl + PageDown / PageUp
Close Tab	⌘ + W	Ctrl + W
Undo	⌘ + Z	Ctrl + Z
Redo	⌘ + ⇧ + Z	Ctrl + Y

More Support

You can navigate to the file in your system explorer

- right click the file then go to Show In -> System Explorer



More Eclipse Resources

- Java Debugging with Eclipse - Tutorial
- Eclipse Help
- SonarSource Java Rules lists over 600 Java rules used in industry to detect bugs, security vulnerabilities, and maintainability issues.