

Announcements

- Participation Assignment in today's class
- Install VSCode, JDK and Maven for the assignment (and for project in general)
- Instructions in this slide deck uploaded on the class website

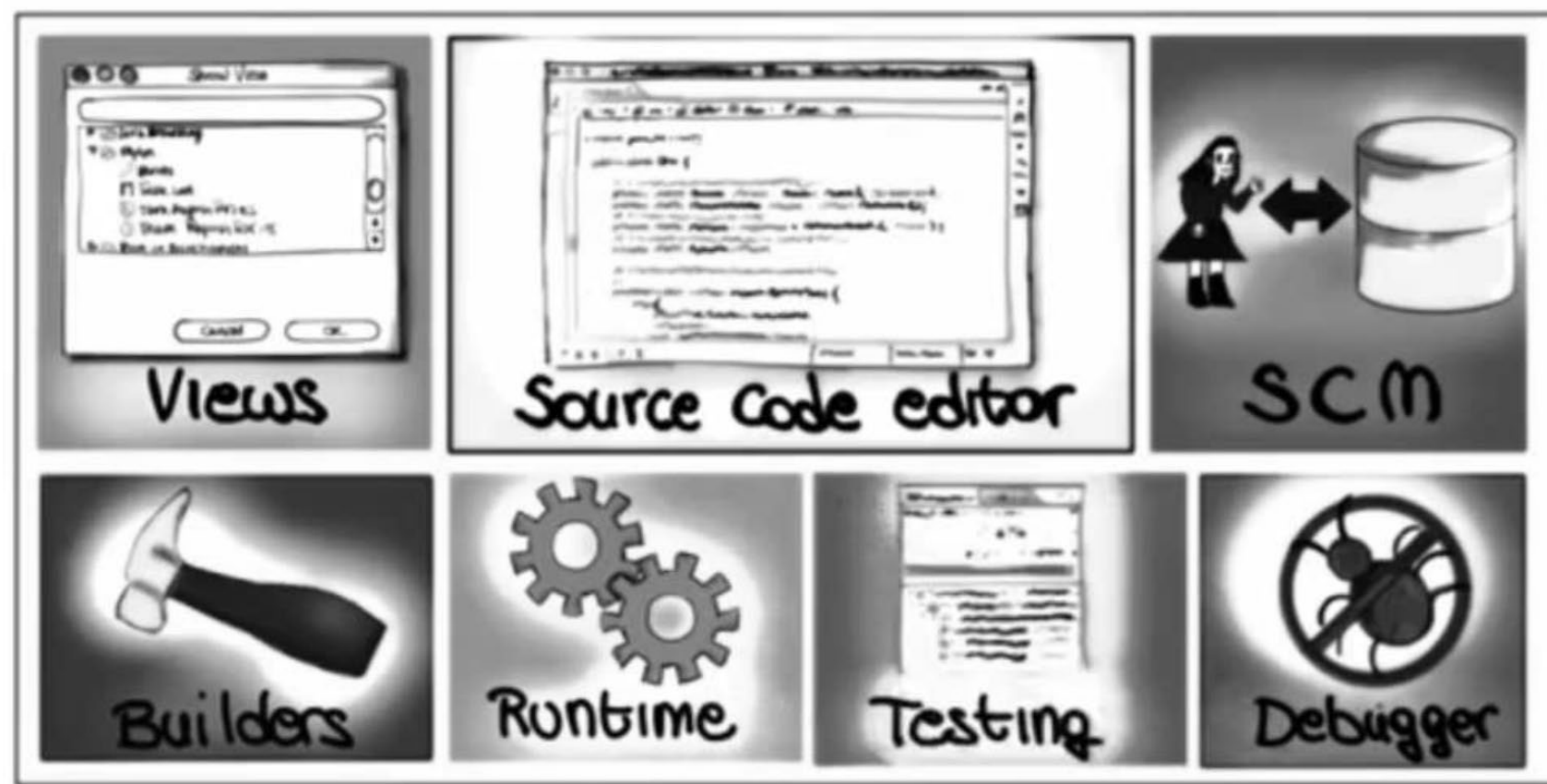
CS3300 Introduction to Software Engineering

Lecture 05: Tools of the Trade #2

VS Code IDE, Junit Testing, Maven

Dr. Nimisha Roy ▶ nroy9@gatech.edu

What is an IDE?



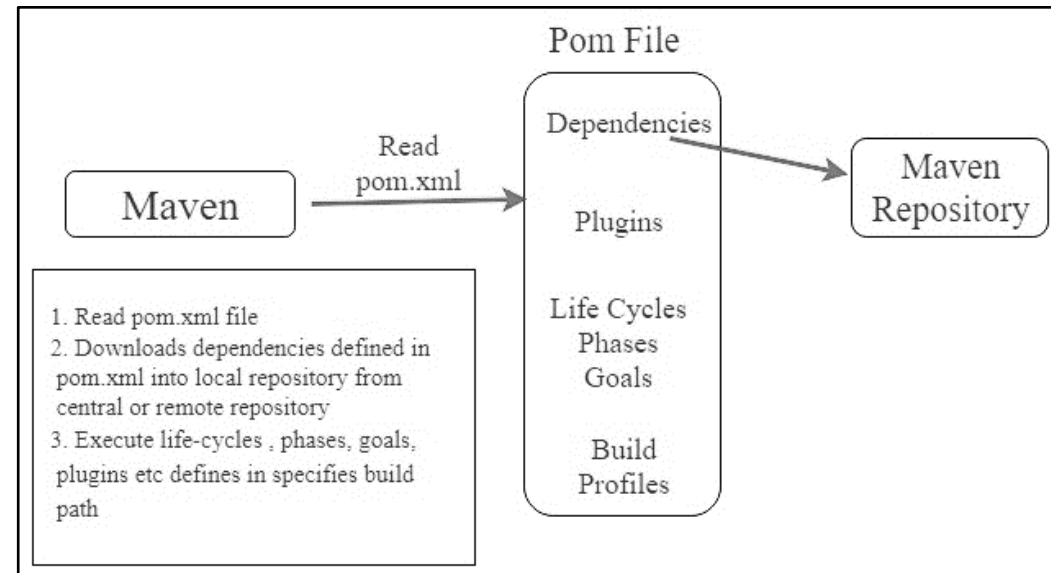
- Integrated Development Environments (IDEs) are software applications that support developers in many of their everyday tasks, such as writing, compiling, and debugging code.
- Some IDEs are designed to support only one programming language such as Java, while others can be used for various languages
- Most popular Java IDEs: VSCode, IntelliJ IDEA, Eclipse, Netbeans

What is an IDE?- VSCode

- VS Code is a lightweight, open-source IDE developed by Microsoft.
- It is highly versatile and supports multiple languages, including Java, Python, JavaScript, and C++.
- Plug-ins provide additional functionality to VS Code, allowing developers to customize the IDE to suit their specific needs. For example, the "Extension Pack for Java" adds Java support, and there are many other plugins for various programming languages and tools, such as Git, Docker, and more.
- VS Code is available on all major operating systems: Mac, Windows, and Linux.

Maven

- Powerful build management tool that can be used for building and managing any Java based project.
 - Easy building of project
 - Generate source code, generate documentation from source code (log document, dependency list, unit test reports etc)
 - Getting right dependencies for project, Compiling source code
 - Packaging compiled codes into JAR, WAR files without scripting
 - Installing packaged code in local, server or central repositories
- Based on POM (Project Object Model)
 - POM files are XML files that contain information related to the project and configuration information such as dependencies, source directory, plugin, goals etc. used by Maven to build the project.



JUnit Testing

- Lightweight framework for creating repeatable tests for your application
- Unit testing in Java
 - Imposes developers' discipline
 - Provides incremental specification
 - Avoids regression errors
 - Allows for changing with confidence
- Very helpful in Test driven development

The JUnit logo features a blue circle on the left, followed by the text 'JUnit' in a serif font. The 'J' is green, the 'U' is red, and 'nit' is also red. The background includes decorative elements: a green triangle pointing right, a blue circle, a yellow vertical bar, and a yellow sun-like shape with an orange semi-circle at the bottom.

JUnit Best Practices

- Keep test cases separate from source code (src/main/tests)
- Method names need to be specific
- Keeping tests simple and focused on one feature or expected result
- Making sure to test for edge cases and not only “perfect scenarios”
- Use appropriate assertions to verify what's expected and what the function returns (actual)
- Tests are repeatable and reliable

1. *assertEquals*(expected, actual): Verifies that the expected and the actual values are equal.

2. *assertNotEquals*(first, second): Verifies that two values are not equal.

3. *assertTrue*(condition): Verifies that the condition is true.

4. *assertFalse*(condition): Verifies that the condition is false.

5. *assertNull*(object): Verifies that the object is null.

6. *assertNotNull*(object): Verifies that the object is not null.

Demo – Run unit tests with Maven in VSCode

- **Install JDK:** Download and install OpenJDK.
- **Set Up VSCode:** Install VSCode and Java Extension Pack.
- **Create Java Project:** Set up a new Java project in VSCode.
- **Add and Run Java Classes:** Create and run simple Java classes.
- **Configure Debugging:** Set up launch.json for custom debugging.
- **Install Maven**
- **Create Maven Project:** Initialize and configure a Maven project.
- **Manage Dependencies:** Edit pom.xml to add dependencies like JUnit.
- **Write and Run JUnit Tests:** Develop and execute unit tests.
- **Demonstrate Debugging:** Use breakpoints and debug features.

Demo – Run unit tests with Maven in VSCode

How to Install JDK

1. Browse to the [AdoptOpenJDK](https://adoptopenjdk.org/) website.
2. Choose your **Operating System**
3. Package type is **JDK**
4. Choose **OpenJDK 16** or the latest version
5. Click the **Latest release** download button to download the package.

How to Install and setup VSCode

1. Browse to <https://code.visualstudio.com/download>
2. Select Download based on your device type
3. Follow the prompted steps and continue
4. Once installed, open VS Code and install the **Extension Pack for Java** from the Extensions Marketplace. This pack includes the essential tools for Java development in VS Code.

Reference: <https://code.visualstudio.com/docs/java/java-tutorial>. You can also download the coding pack and the extension pack instead from the above link.

Demo – Run unit tests with Maven in VSCode

How to Install Maven

1. Browse to the [Apache Maven Project Downloads page](#).
2. Install Maven
3. Add Maven to the PATH environment variable
4. Set JAVA_HOME (if not already set):

Demo – Run Configurations

By default using launch.json. Doesn't need to be touched unless:

- Custom Run Configurations: If you have specific requirements for launching your application, such as setting environment variables, JVM arguments, or choosing among multiple main classes, you can define these configurations in launch.json.
- Debugging with Specific Parameters: If your application requires command line arguments to run, you can specify these in the launch.json file. This is useful when your program expects user input that would typically be provided when running the program from a command line.
- Complex Build and Launch Processes: For applications that require more complex build setups or pre-launch tasks (like compiling resources other than Java files, moving files, etc.), you can define pre-launch tasks in launch.json that execute these commands before debugging starts.
- Remote Debugging: When you need to connect to a Java application running on a different machine or environment, you can configure the necessary remote debugging settings in launch.json.

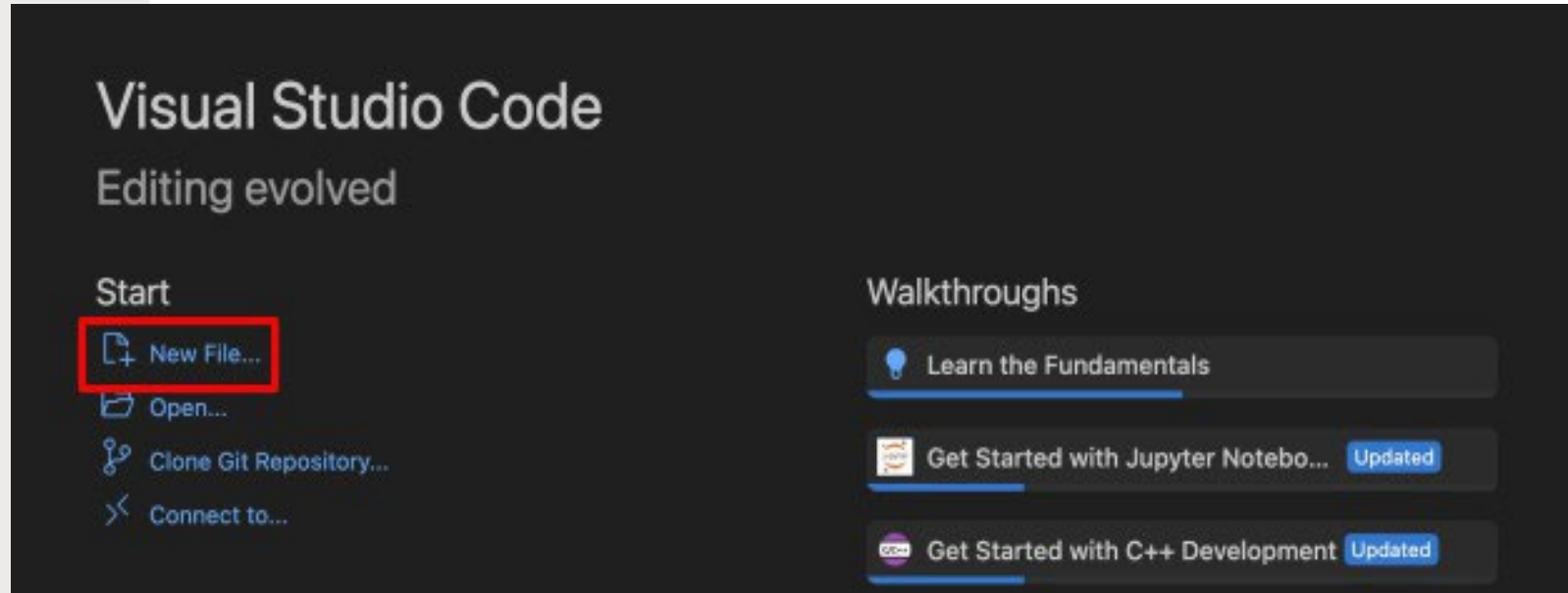
Debug:

Add breakpoints – conditional breakpoint; data breakpoint; logpoints

Demo – create project

- How to create a new project with Maven

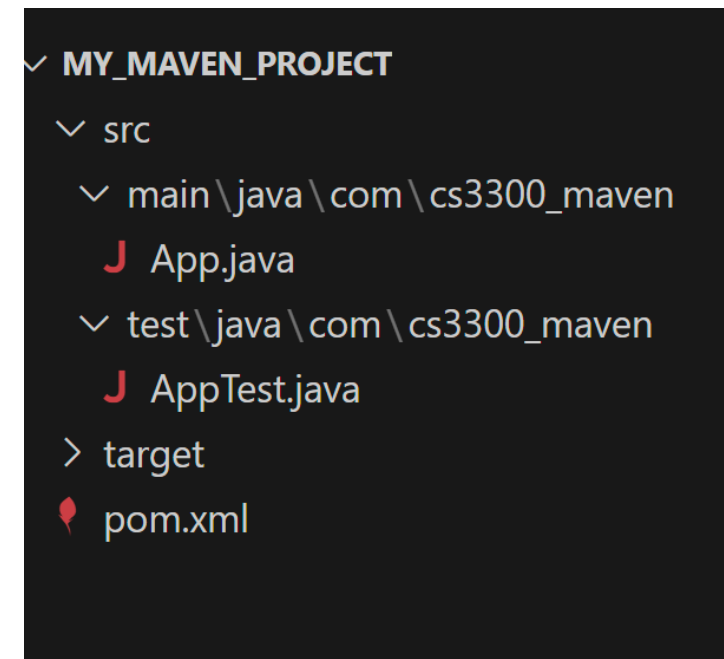
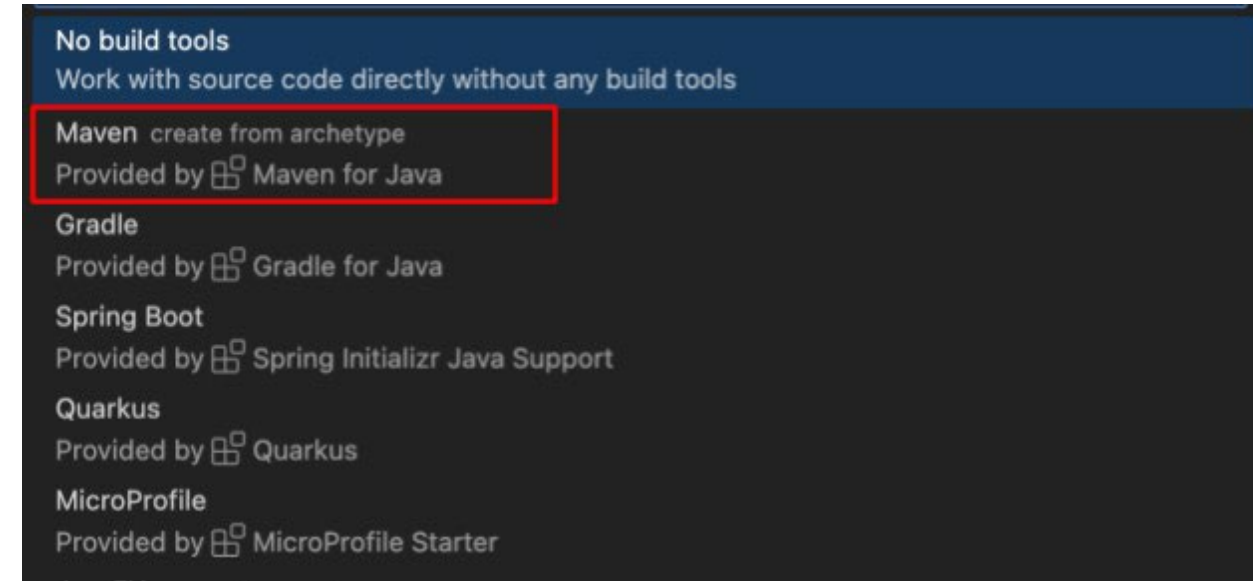
- Select New File
- Search Java Project in the search bar and click New Java Project
- Select No Build Tools and Name the project myFirstProject
- Open the Command Palette (Ctrl + Shift + P or Cmd + Shift + P)
- Search for Java: Configure Java Runtime and select it. Ensure that the JDK downloaded before is the one selected



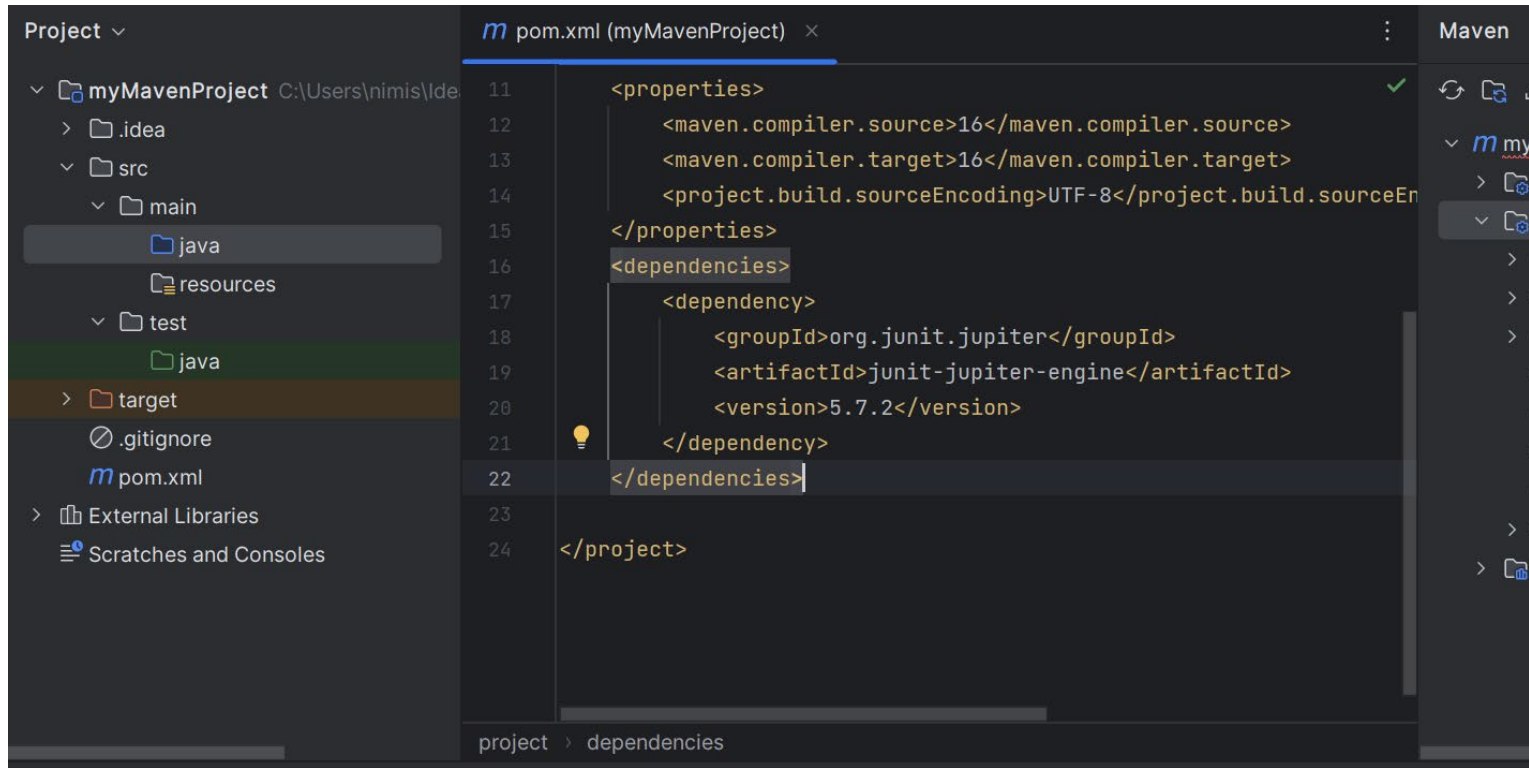
Demo – create project

How to create a project with Maven

- Choose groupid and artifactid
 - Groupid: uniquely identifies your project across all projects. Kind of like package. *com.cs3300_maven*
 - artifactid is the name of the project. Let's say *my_maven_project*
- You will be able to see pom.xml file in the package explorer



Demo- Editing pom.xml to add dependency



```
11 <properties>
12     <maven.compiler.source>16</maven.compiler.source>
13     <maven.compiler.target>16</maven.compiler.target>
14     <project.build.sourceEncoding>UTF-8</project.build.sourceEn
15 </properties>
16 <dependencies>
17     <dependency>
18         <groupId>org.junit.jupiter</groupId>
19         <artifactId>junit-jupiter-engine</artifactId>
20         <version>5.7.2</version>
21     </dependency>
22 </dependencies>
23
24 </project>
```

Edit pom.xml to add Junit as a dependency

1. In **pom.xml**, press **Alt + Insert** and select **Dependency**.

2. In the dialog that opens, type **org.junit.jupiter:junit-jupiter** in the search field. Locate the necessary dependency in the search results and click **Add**.

Demo- Adding code and test cases

- Create **class** in src/main/java. *AddConcatenate.java*
- Add 2 methods, 1 to add 2 numbers and 1 to concatenate 2 strings.
- Create corresponding 2 junit tests to test the 2 functions
- Run tests

What would be good test cases here?

- Add: addition with 0, addition with negative numbers, addition resulting in 0, addition with MAX INTEGER and MIN INTEGER (underflow and overflow)
- Concat: Concatenate with an Empty String, concatenate Special Characters, Concatenate with Numbers, Concatenate Two Empty Strings, concatenate non English characters

Participation Assignment