Java: From Core Concepts to Full-Stack Mastery

Session 1: Welcome & Environment Setup

Why Invest in Java Now?

For a developer with front-end expertise, Java is the key to unlocking full-stack capabilities and building robust, enterprise-grade applications.

Evergreen Technology

An industry pillar, not a fleeting trend. Essential for long-term career stability.

The Enterprise Standard

The backbone of critical industries like banking and e-commerce, trusted by clients like Mastercard and Citi.

Secure & Robust

Trusted for mission-critical systems that demand the highest levels of performance and security.

High Demand

Directly connects learning Java to strong, immediate career prospects in the current market.

Your Path to Full-Stack Proficiency

1

Core Java (The Foundation)

Master the language fundamentals. This is non-negotiable. The stronger your Core Java, the better you can focus on Spring Boot.

2

Spring Boot (The Framework)

Learn to build powerful, production-grade applications rapidly. This is what the industry uses.

3

Microservices (The Architecture)

Design scalable, resilient systems for modern applications. This is an architectural concept built on top of your Spring Boot skills.

From Human Logic to Machine Language

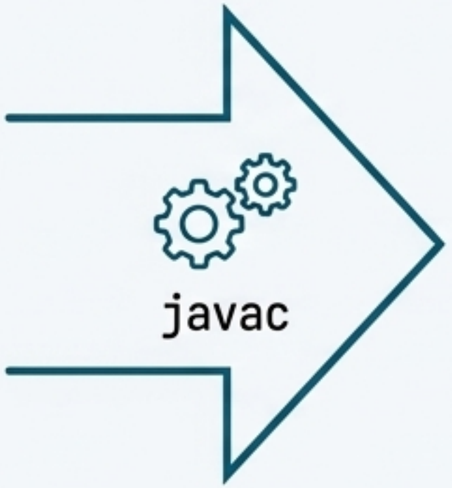
The "write once, run anywhere" philosophy is powered by a two-step process: writing **human-readable code** and compiling it into **universal machine-readable code**.

Source Code (`.java` file)

Human-readable instructions that you, the developer, will write.

HelloWorld.java

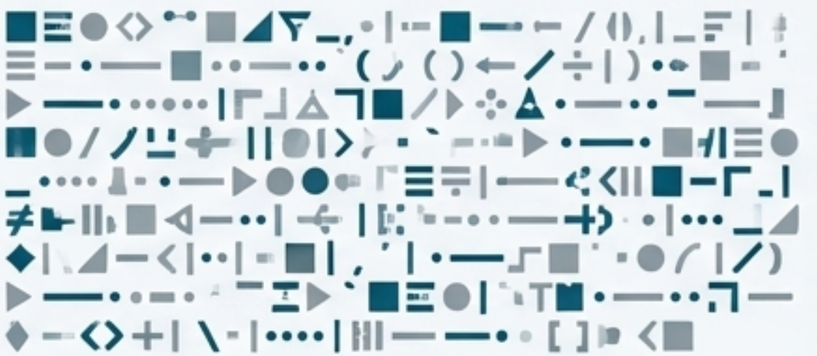
```
public class HelloWorld {
  public sen HelloWorld {
  public static void main(String[] args) {
    System.out.println("Hello, World!");
  }
}
```



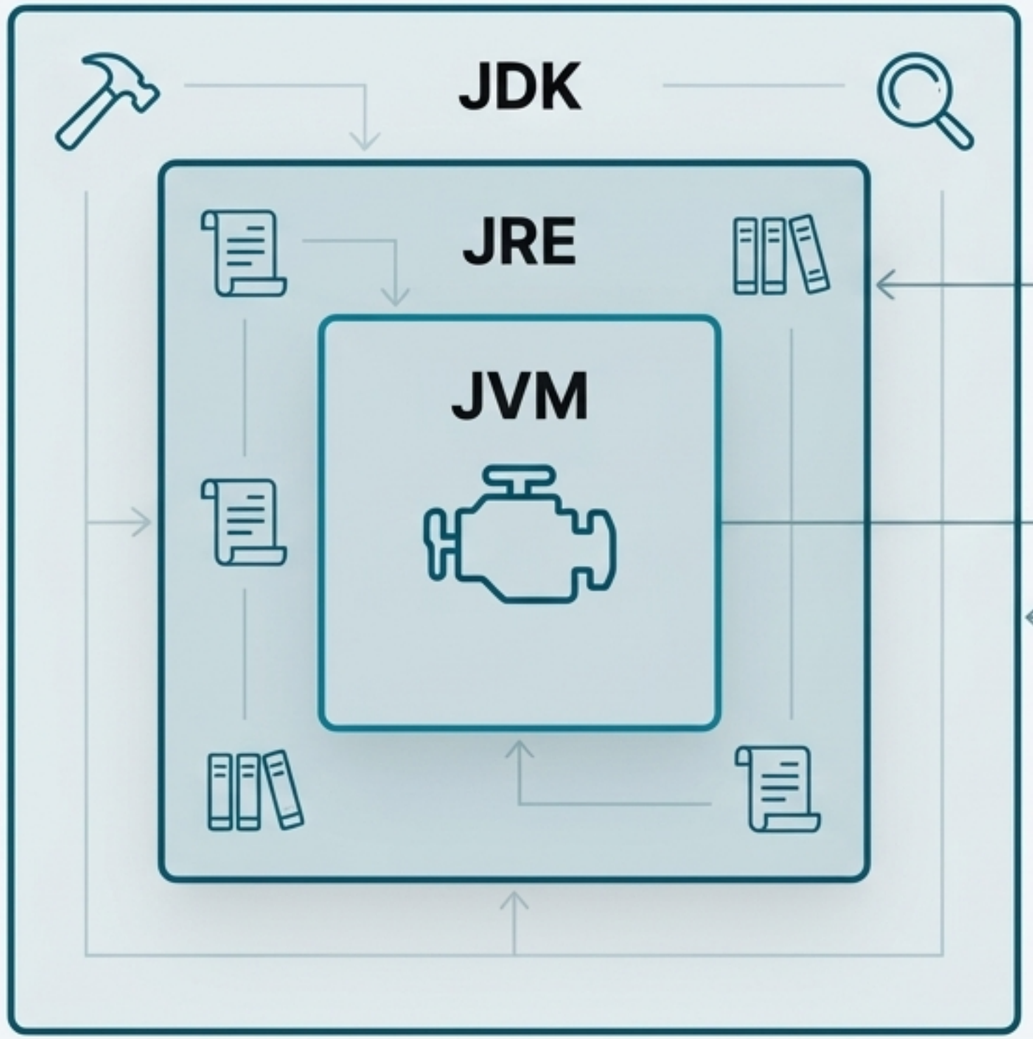
Bytecode (`.class` file)

A compiled, platform-independent version of your code. It's the "universal language" that any Java Virtual Machine (JVM) can understand and execute.

HelloWorld.class



Inside the Java Platform: JDK, JRE & JVM



JVM (Java Virtual Machine)

- **Role:** The core engine that executes the bytecode (`.class` files). It's the "virtual computer."

JRE (Java Runtime Environment)

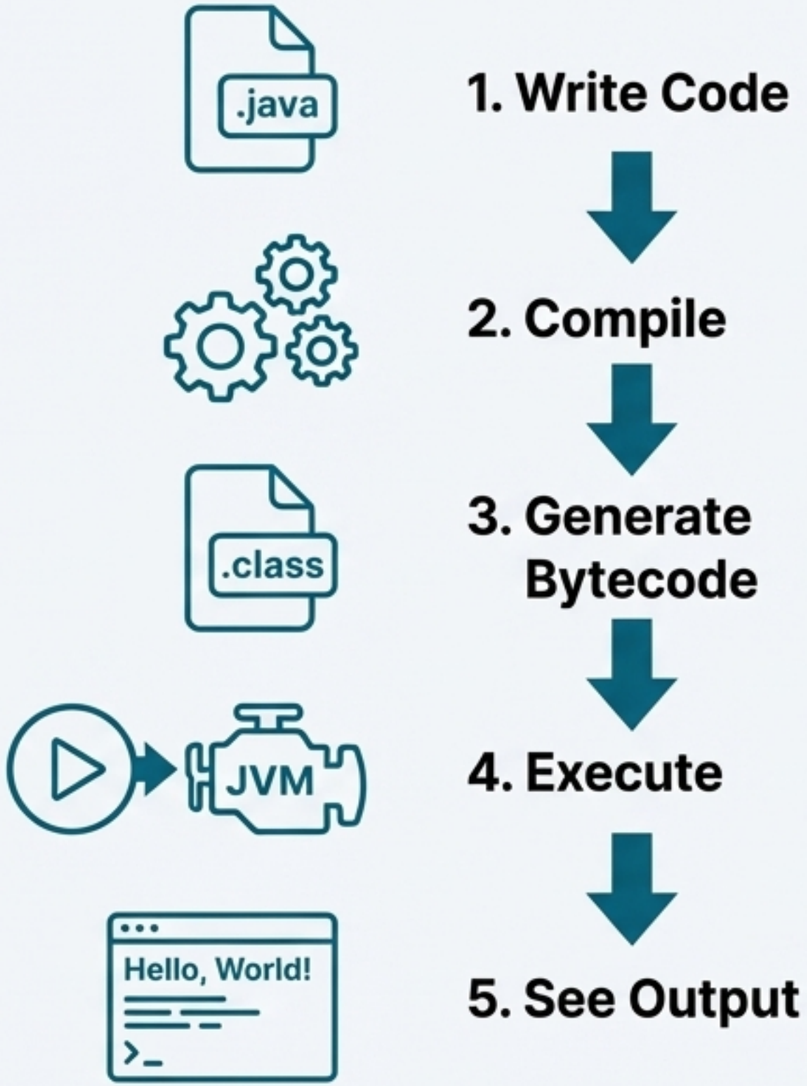
- **Formula:** JVM + Core Libraries
- **Role:** Everything needed to *run* a pre-compiled Java application. Servers deploying applications only need the JRE.

JDK (Java Development Kit)

- **Formula:** JRE + Development Tools (Compiler, Debugger)
- **Role:** The complete toolkit needed to *write, compile, and build* Java applications.

As a developer, you need the JDK.

How Your Code Comes to Life



1. Write Code

You create your program in a text file named YourProgram.java.

2. Compile

You use the Java Compiler from the JDK by running the command: `javac YourProgram.java`.

3. Generate Bytecode

The compiler successfully creates the YourProgram.class file (bytecode).

4. Execute

You use the Java launcher from the JRE by running the command: `java YourProgram`. This starts the JVM.

5. See Output

The JVM interprets the bytecode and produces the final output in your console.

The Successful Developer's Mindset

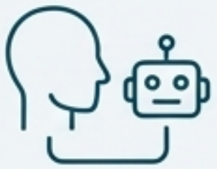
Technology is only half the battle. Your approach to learning and problem-solving is what will truly define your success.

Practice Daily

"Avoid carrying work forward." Consistent, daily practice is crucial as the syllabus is vast.

Embrace Debugging

It is a core skill for **understanding* code flow, not a sign of failure. Get comfortable using the debugger from day one.



Use AI as a Co-pilot, Not a Crutch

Leverage tools to assist, but never to bypass understanding. "If you use shortcuts, your logic will never build."

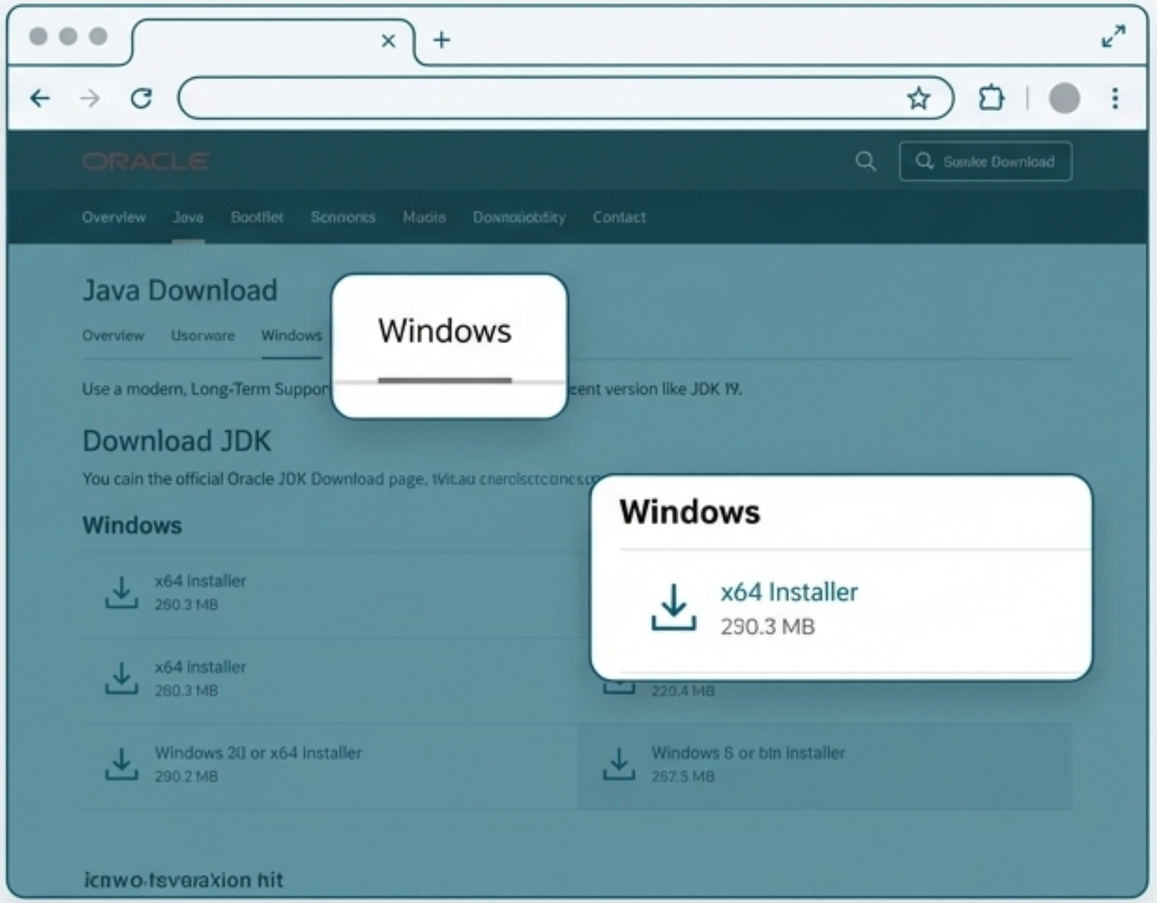
Ask Questions

"No question is too small." An interactive and curious mindset is essential for learning.

Step 1: Install the Java Development Kit (JDK)

Use a modern, Long-Term Support (LTS) version like JDK 21, or a recent version like JDK 19.

1. Go to the official Oracle JDK Download page.
2. Select the **Windows** tab and download the **x64 Installer**.
3. Run the installer and follow the on-screen prompts (default settings are fine).

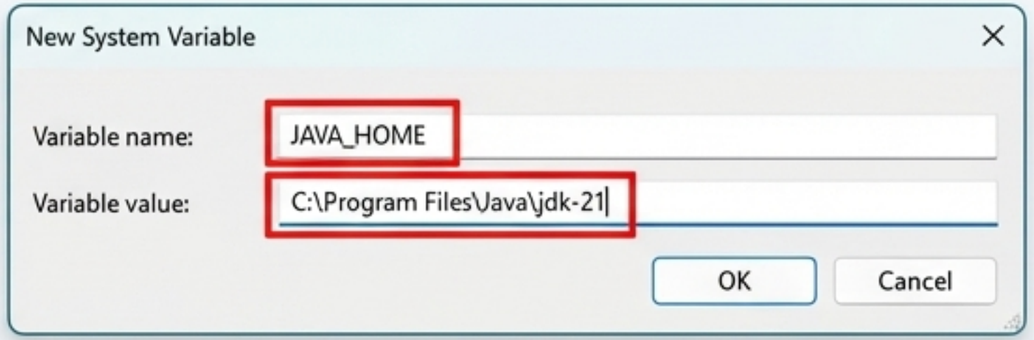


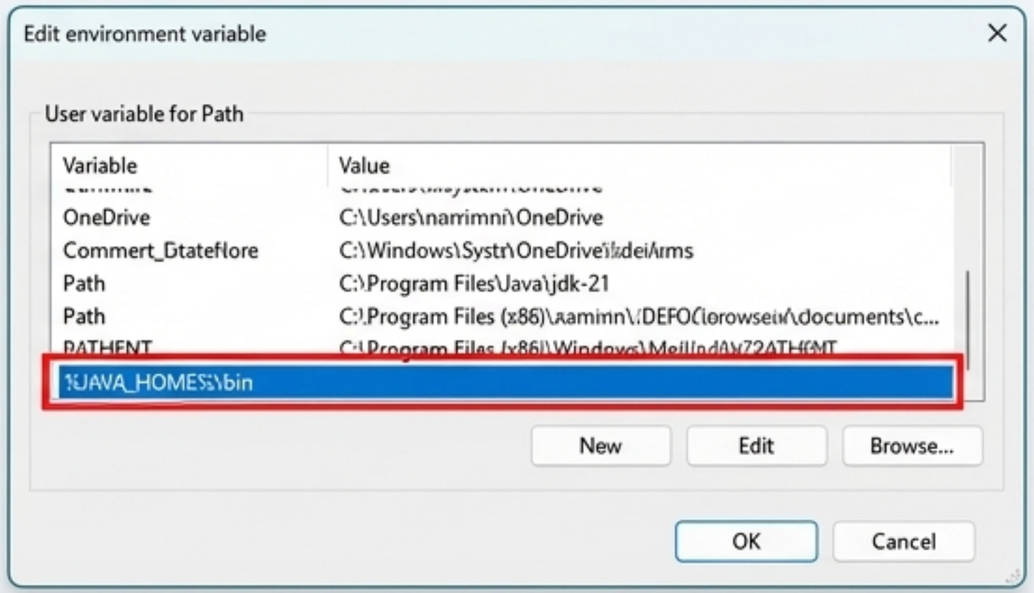
Step 2: Configure Your Environment Variables

This step tells your operating system where to find Java commands (like `java` and `javac`) from any location.

Windows

1. Search for "**Edit the system environment variables**" and open it.
2. Click the "**Environment Variables...**" button.
3. Under **System variables**, click "**New...**":
   - Variable name: `JAVA_HOME`
   - Variable value: `C:\Program Files\Javajdk-21` (or your specific installation path)
4. Find the **Path** variable, select it, and click "Edit...". Add a new entry: `%JAVA_HOME%\bin`

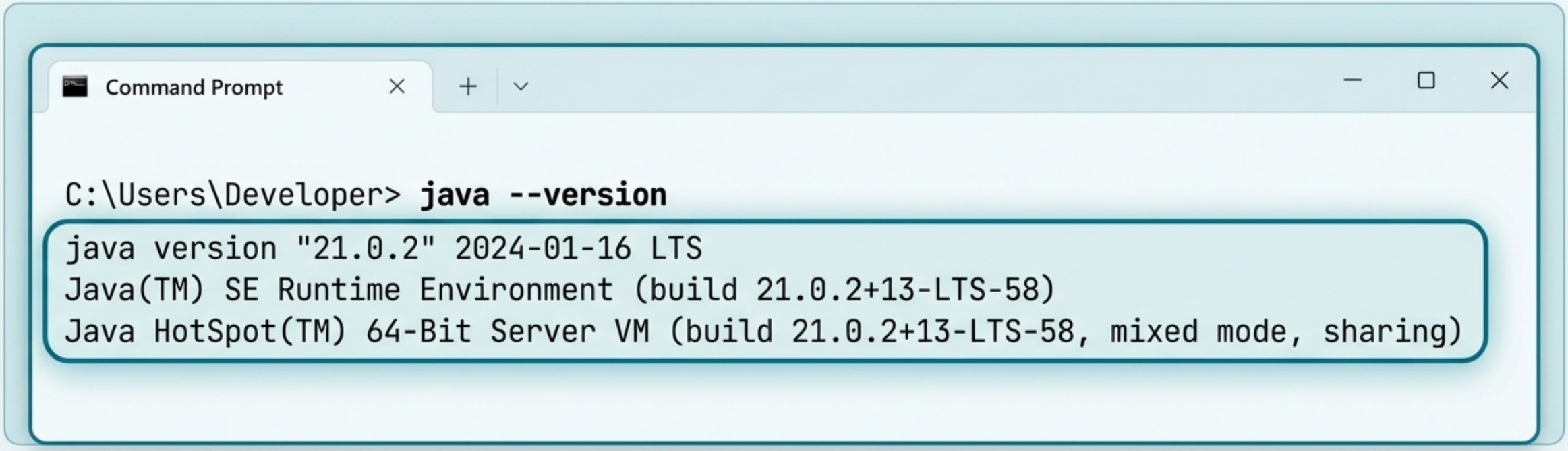




Step 3: Verify Your Installation

Let’s confirm that your system recognizes the new Java installation.

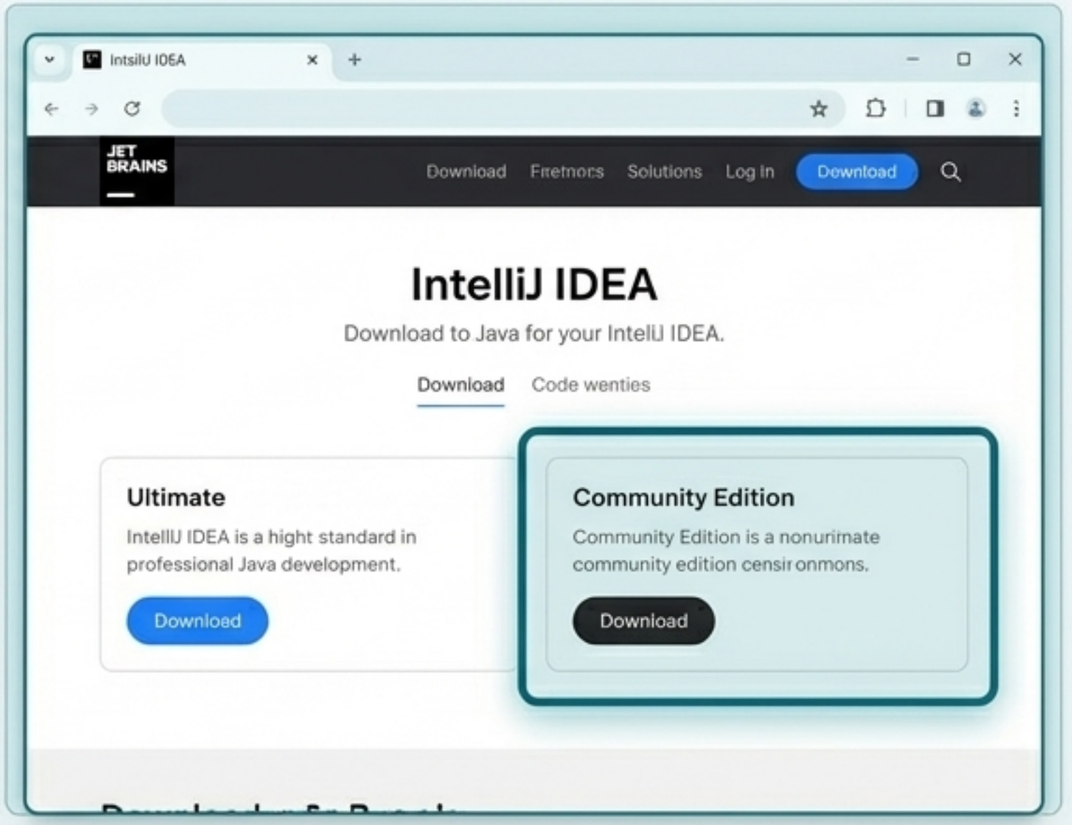
1. Open a new Command Prompt (CMD) or Terminal window.
2. Type the following command and press Enter: `java --version`



Step 4: Install Your Integrated Development Environment (IDE)

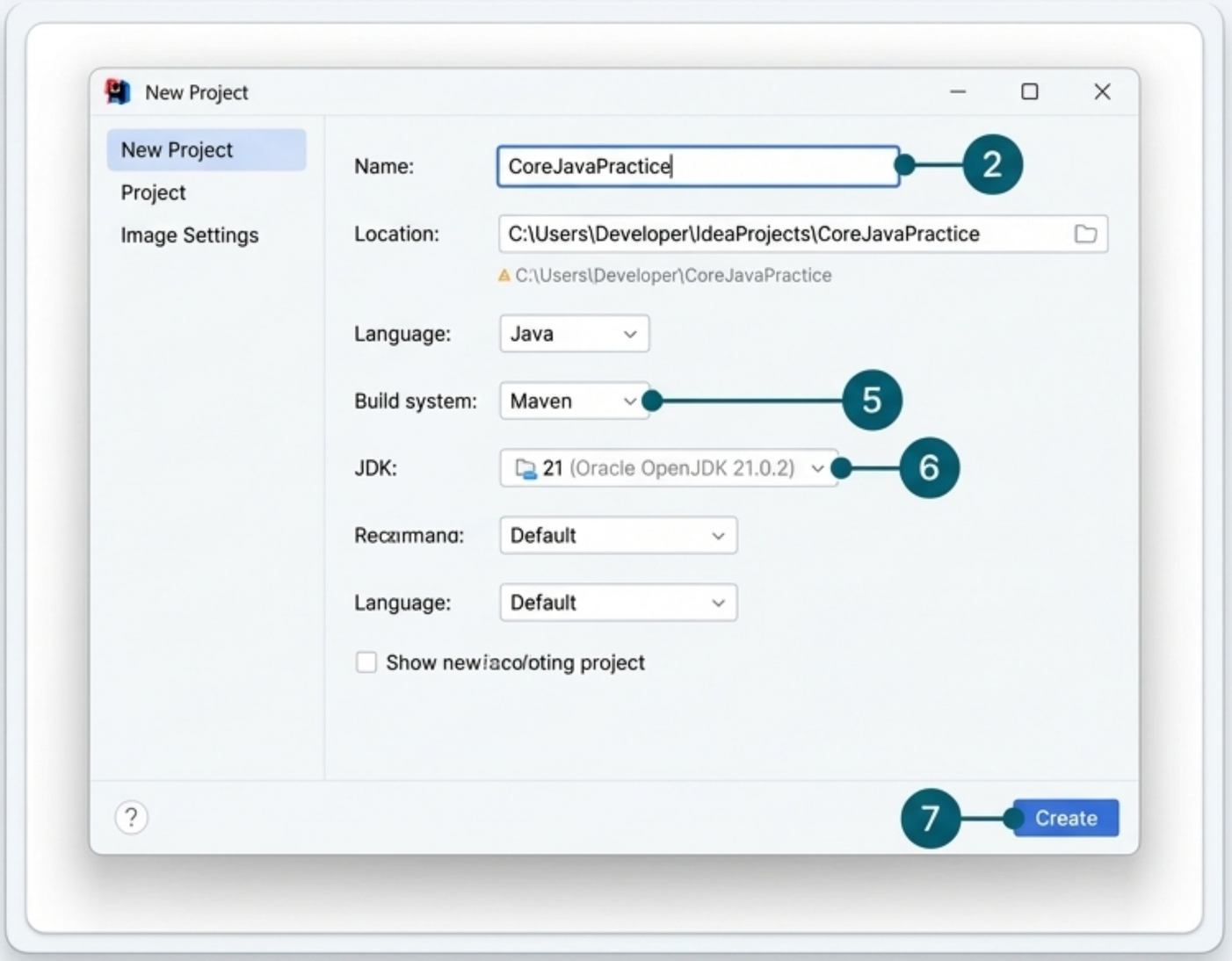
We will use **IntelliJ IDEA**. It is a dominant standard in professional Java development. The free **Community Edition** is all you need to start.

1. Go to the JetBrains IntelliJ IDEA download page.
2. Download the **Community Edition** installer.
3. Run the installer. The default options are sufficient for now.



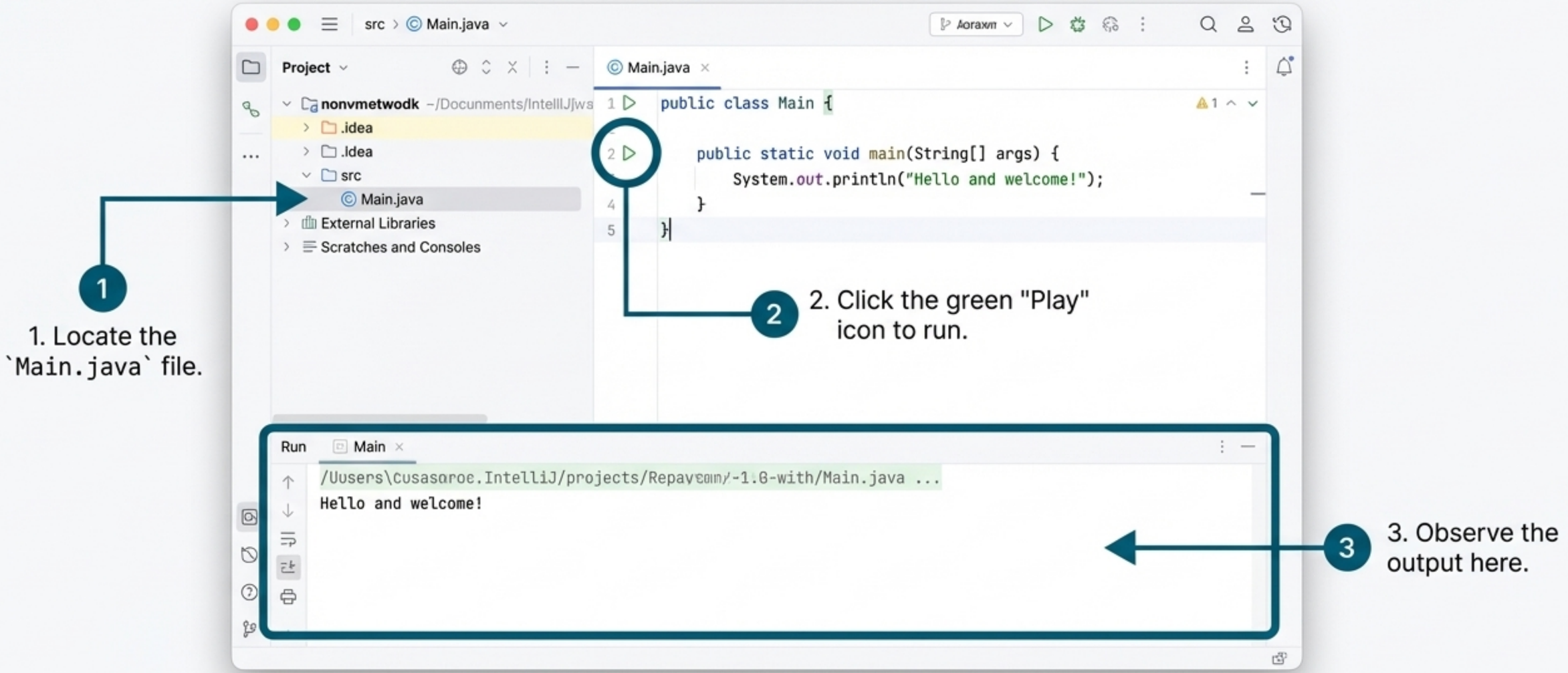
Bringing It All Together: Your First Project

1. Open IntelliJ and select **File -> New -> Project...**
2. **Name** your project (e.g., `CoreJavaPractice`).
3. **Location**: Keep the default or choose a preferred folder.
4. **Language**: Java
5. **Build system**: Select **Maven**.
6. **JDK**: Ensure your newly installed JDK (e.g., version 21) is selected. `JetBrains`
7. Click **Create**.



From Code to Console: Running Your First Program

IntelliJ automatically creates a `Main.java` file with a basic 'Hello, World!' program. Let's run it.



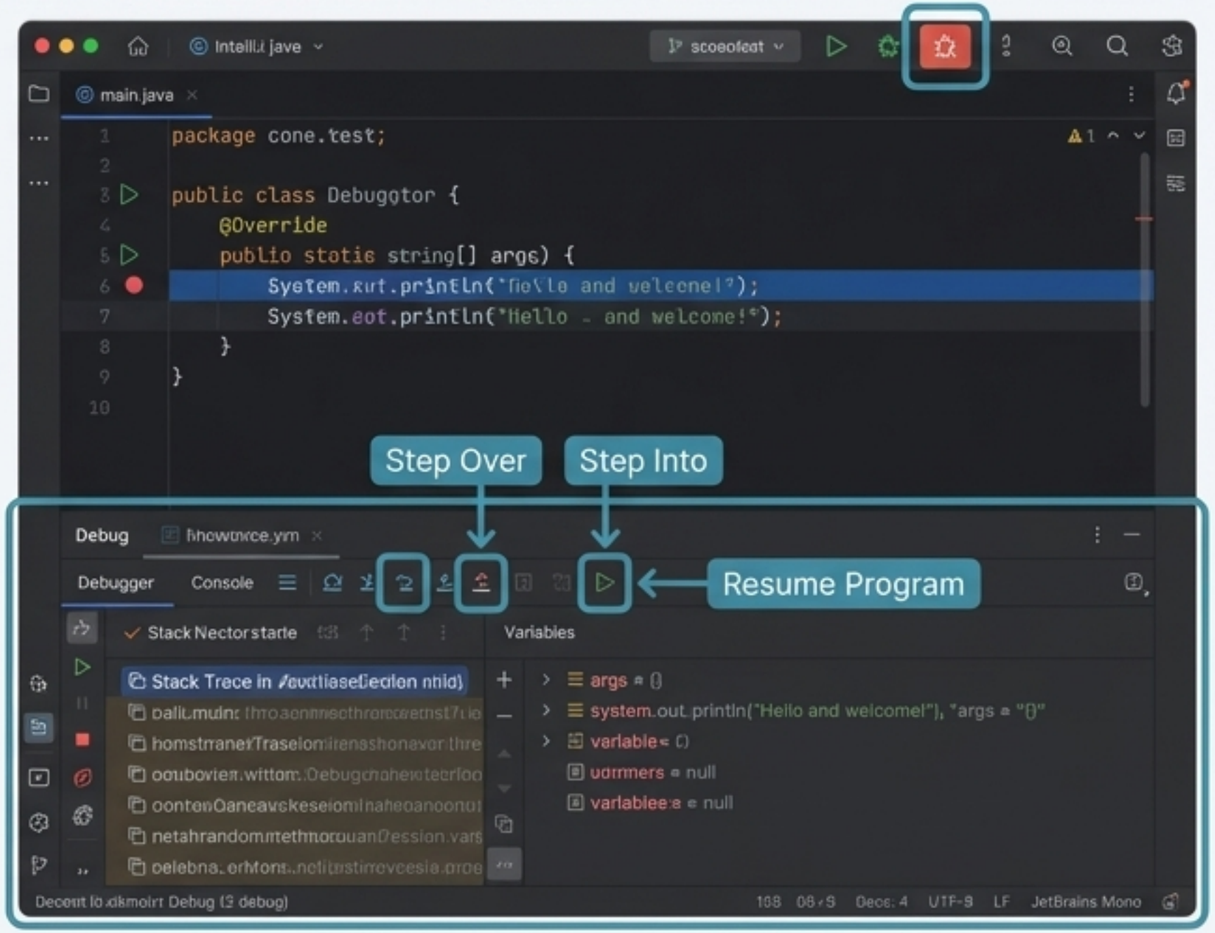
Beyond 'Run': A Glimpse into Debugging

Core Idea: The debugger is the most powerful tool for understanding code flow. It allows you to pause execution and inspect the state of your program line by line.

1. Set a Breakpoint: Click in the empty space (the "gutter") to the left of a line number. A red dot will appear.

2. Run in Debug Mode: Instead of the "Play" icon, click the "Bug" icon next to it.

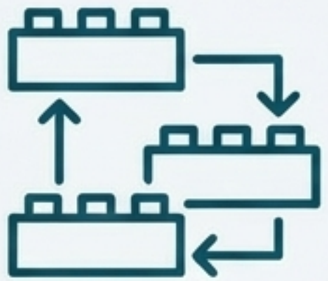
3. Step Through Code: When the program pauses, use the control buttons in the Debug window to execute one line at a time.



The Journey Ahead

Next Session

We will dive into the building blocks of the Java language: **Data Types, Variables, and Control Flow Statements**.



Your Task Before Next Class

- ☑ Ensure your JDK and IntelliJ IDEA are fully installed.
- ☑ Successfully create a new Maven project in IntelliJ.
- ☑ Run the default `Main.java` file and see the "Hello, World!" output.

The foundation we build now will support every complex application you create in the future. Let's get building.