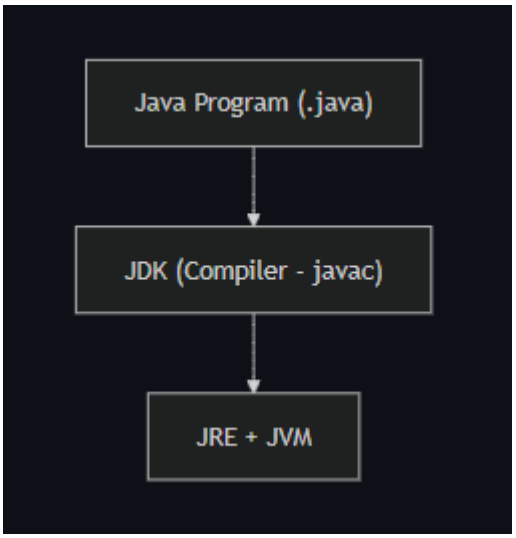
Lesson Notes: 02/02/2026

Java Ecosystem



The JDK (Java Development Kit) is the full toolkit needed to develop Java programs. It contains:

- The Compiler (javac) which compiles the Java bytecode
- The JRE (Java Runtime Environment)
- Developer tools (debuggers, profilers, etc.)

The JRE (Java Runtime Environment) is the environment required to run Java programs. It includes:

- The JVM (Java Virtual Machine)
- Core libraries (e.g., java.lang, java.util)

The JVM (Java Virtual Machine) is the engine that runs Java bytecode, interpreting or JIT-compiling to machine code, regardless of the underlying hardware/OS on which it is installed.

The same .class file can run on Windows, macOS, or Linux because each has its own JVM implementation.

To compile a program from the command line:
```
javac -d .\target\classes\ .\src\main\java\com\sparta\setup\HelloWorldApp.java
```

And then to run it:
```
java -cp .\target\classes\ com.sparta.setup.HelloWorldApp
```

Running it with arguments:
```
java -cp .\target\classes\ com.sparta.setup.HelloWorldApp Phil
```

Static main

Part	Meaning
`public`	Accessible to the JVM from anywhere. The JVM must be able to call it.
`static`	Belongs to the class, not to an instance — the JVM can call it without creating an object.
`void`	The method doesn't return any value to the JVM.
`main`	The predefined name recognized by the JVM as the program's entry point.
`String[] args`	Command-line arguments passed as an array of strings.

Rules for Variables

- Must start with a letter or an underscore (_)
    - Cannot start with a number
- Cannot contain any spaces
- Names are case-sensitive (age ≠ Age)
- Cannot be a Java keyword (int, class, etc.)

Conventions and Good Practice

Follow the Google Style Guide but, for now, keep the following in mind:

- Use camelCase for variable and method names, use PascalCase for class names (see Rules by identifier type in the Google Style Guide)
- Use descriptive names, not x or y, unless very short examples

Rules for constants

- Must be given a value when declared

- A constant cannot be reassigned a value

Convention and Good Practice

- By convention, constants are written in (UPPER_SNAKE_CASE)

Different types of primitive

- byte
- short
- int
- long
- float
- double
- char
- boolean

Static Typing vs. Dynamic Typing

While statically typed languages offer the developer peace of mind, they mean that you can't just change the type of a variable on the fly (at runtime).
Some languages are dynamically typed which means that the type is inferred at runtime. Although this is more flexible, you give up the following advantages of a statically typed language:

- Type safety: Errors are caught before you run the program
- Better performance: The compiler knows the exact data types ahead of time, allowing optimizations
- Easier to maintain: You know what type every variable is supposed to be
- Tooling support: IDEs (like IntelliJ or Eclipse) can give better autocomplete and refactoring help

Type conversions

byte → short → int → long → float → double

Escape Sequence	Meaning	Example	Output
`\n`	New line	`"Hello\nWorld"`	`Hello` `World`
`\t`	Tab (horizontal)	`"A\tB\tC"`	`A B C`
`\"`	Double quote	`"He said, \"Hi!\""`	`He said, "Hi!"`
`\\`	Backslash	`"C:\\Program Files\\"`	`C:\Program Files\`

Common placeholders

Placeholder	Description	Example
%d	Integer	10
%f	Floating-point number	3.14
%.2f	Floating-point with 2 decimals	3.14
%s	String	"Hello"
%n	Newline (platform-independent)	