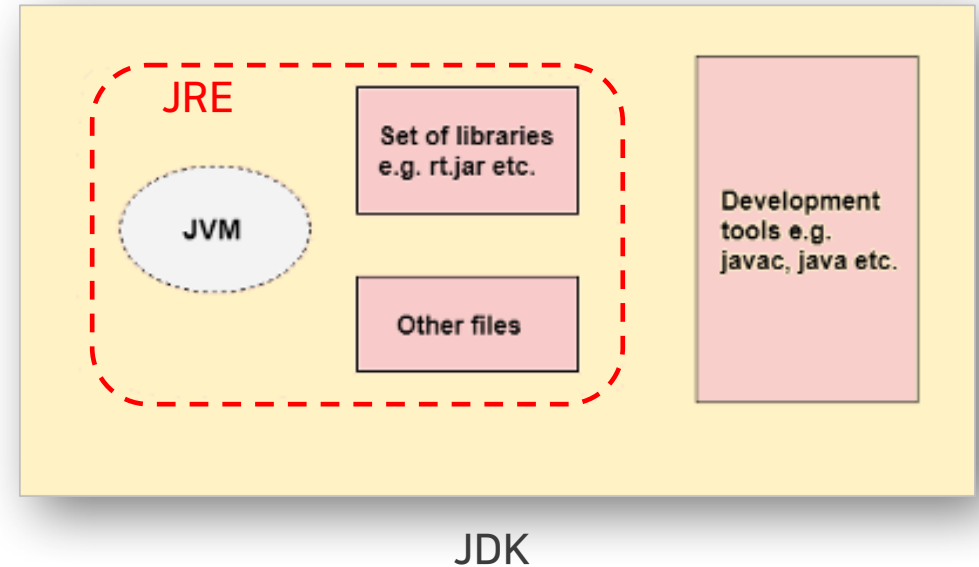


# **01\_3 Environment Setup**

Object-Oriented Programming

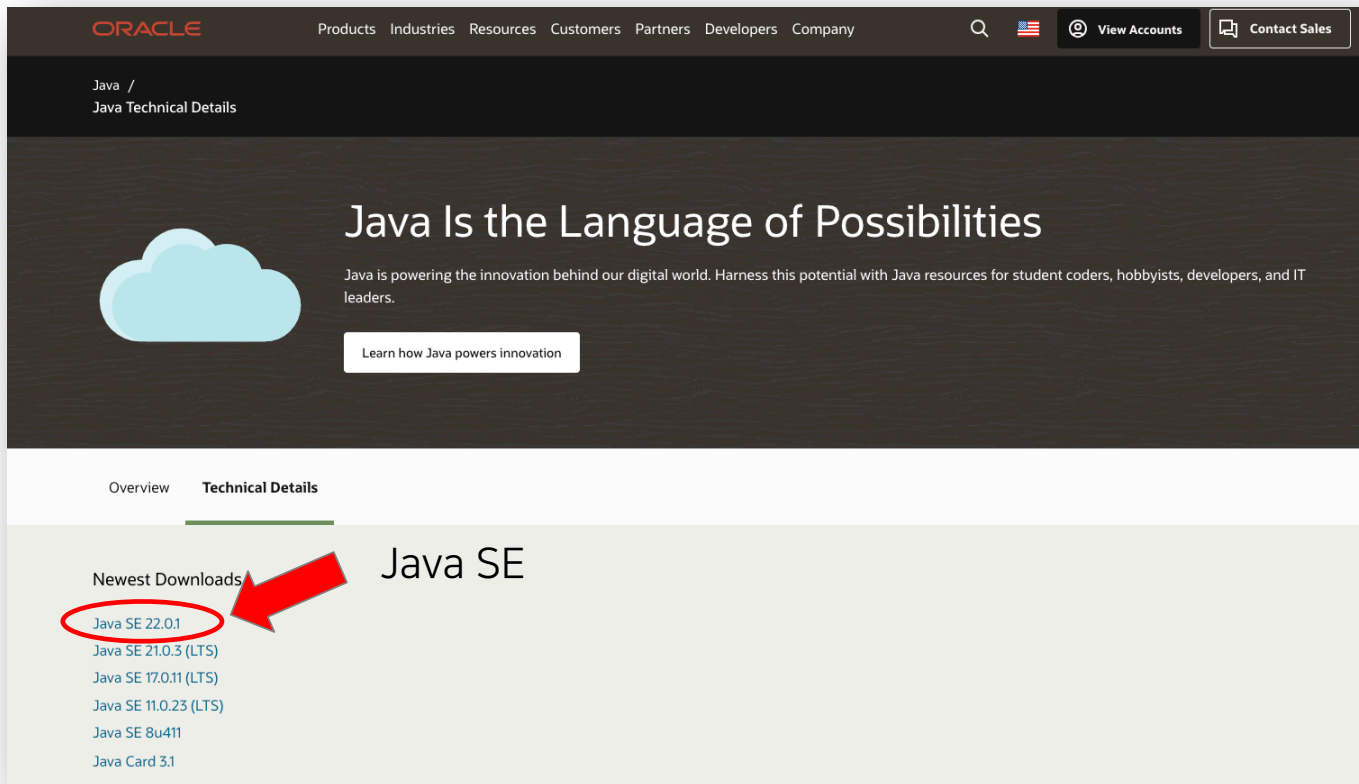
# JDK and JRE

- JDK (Java Development Kit)
  - Including tools for developers
  - JVM, Java Compiler, ...
- JRE (Java Runtime Environment)
  - Execution environment for program users
  - Including JVM but no development tools
  - $JRE \subset JDK$
- Distributing JDK and JRE
  - Oracle's Java Technology Site
    - <https://www.oracle.com/java/technologies/>



# Installing JDK - 1

- <https://www.oracle.com/java/technologies/>



# Installing JDK - 2

- Download appropriate installation file for your platform:
- Windows
  - x64 installer
    - [https://download.oracle.com/java/22/latest/jdk-22\\_windows-x64\\_bin.exe](https://download.oracle.com/java/22/latest/jdk-22_windows-x64_bin.exe)
- macOS
  - For M1, M2, ... (ARM) Chips
    - [https://download.oracle.com/java/22/latest/jdk-22\\_macos-aarch64\\_bin.dmg](https://download.oracle.com/java/22/latest/jdk-22_macos-aarch64_bin.dmg)
  - For Intel Chips:
    - [https://download.oracle.com/java/22/latest/jdk-22\\_macos-x64\\_bin.dmg](https://download.oracle.com/java/22/latest/jdk-22_macos-x64_bin.dmg)

# Checking JDK Installation

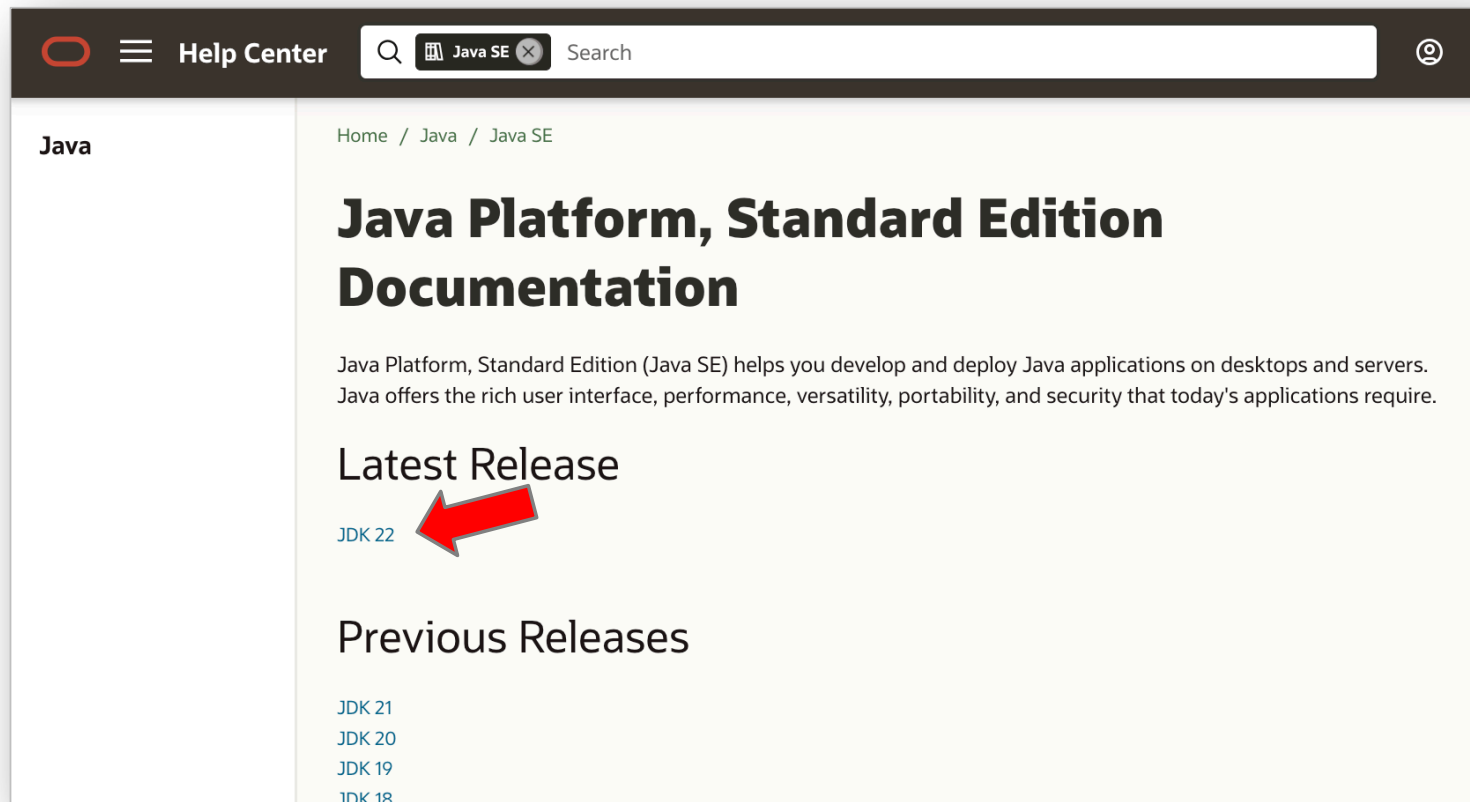
- Executing Console (Command prompt, Terminal)
  - Windows
    - Windows Key + R > type "cmd" > confirm
    - Start menu > Windows system > command prompt
  - Mac: terminal.app
- Type `java -version` and `javac -version` to confirm the installation

```
C:\> java -version
java version "17.0.1" 2021-10-19 LTS

C:\> javac -version
javac 17.0.1
```

# JavaSE Documentation

<https://docs.oracle.com/en/java/javase>



The screenshot displays the Oracle Help Center for Java Platform, Standard Edition (Java SE) documentation. The page has a dark header with the Oracle logo, a hamburger menu, 'Help Center', a search bar containing 'Java SE', and a user icon. A left sidebar shows 'Java' as the selected category. The main content area has a breadcrumb 'Home / Java / Java SE' and a title 'Java Platform, Standard Edition Documentation'. Below the title is a paragraph describing Java SE. The 'Latest Release' section features a red arrow pointing to 'JDK 22'. The 'Previous Releases' section lists 'JDK 21', 'JDK 20', 'JDK 19', and 'JDK 18'.

**Java**

Home / Java / Java SE

## Java Platform, Standard Edition Documentation

Java Platform, Standard Edition (Java SE) helps you develop and deploy Java applications on desktops and servers. Java offers the rich user interface, performance, versatility, portability, and security that today's applications require.

### Latest Release

[JDK 22](#)

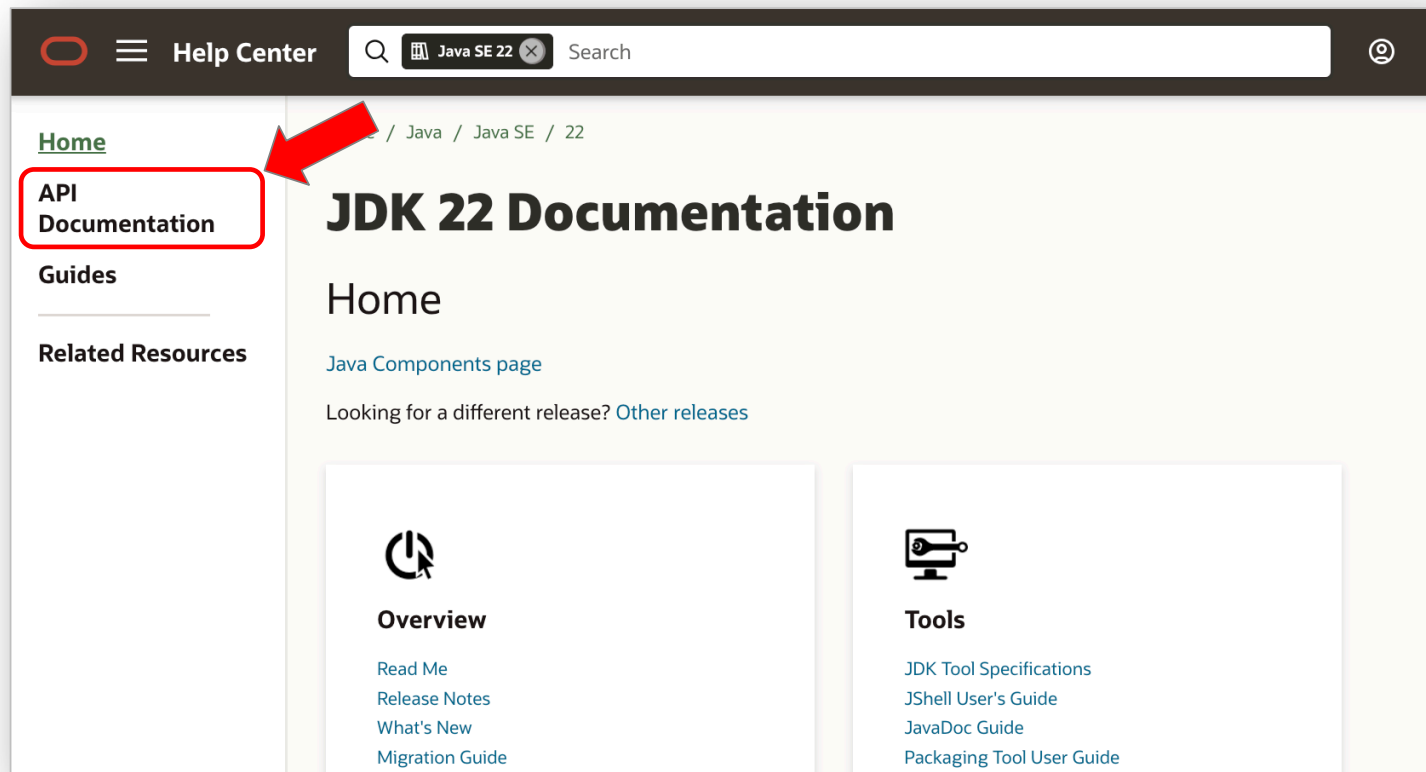
### Previous Releases

- [JDK 21](#)
- [JDK 20](#)
- [JDK 19](#)
- [JDK 18](#)

# JDK Documentation

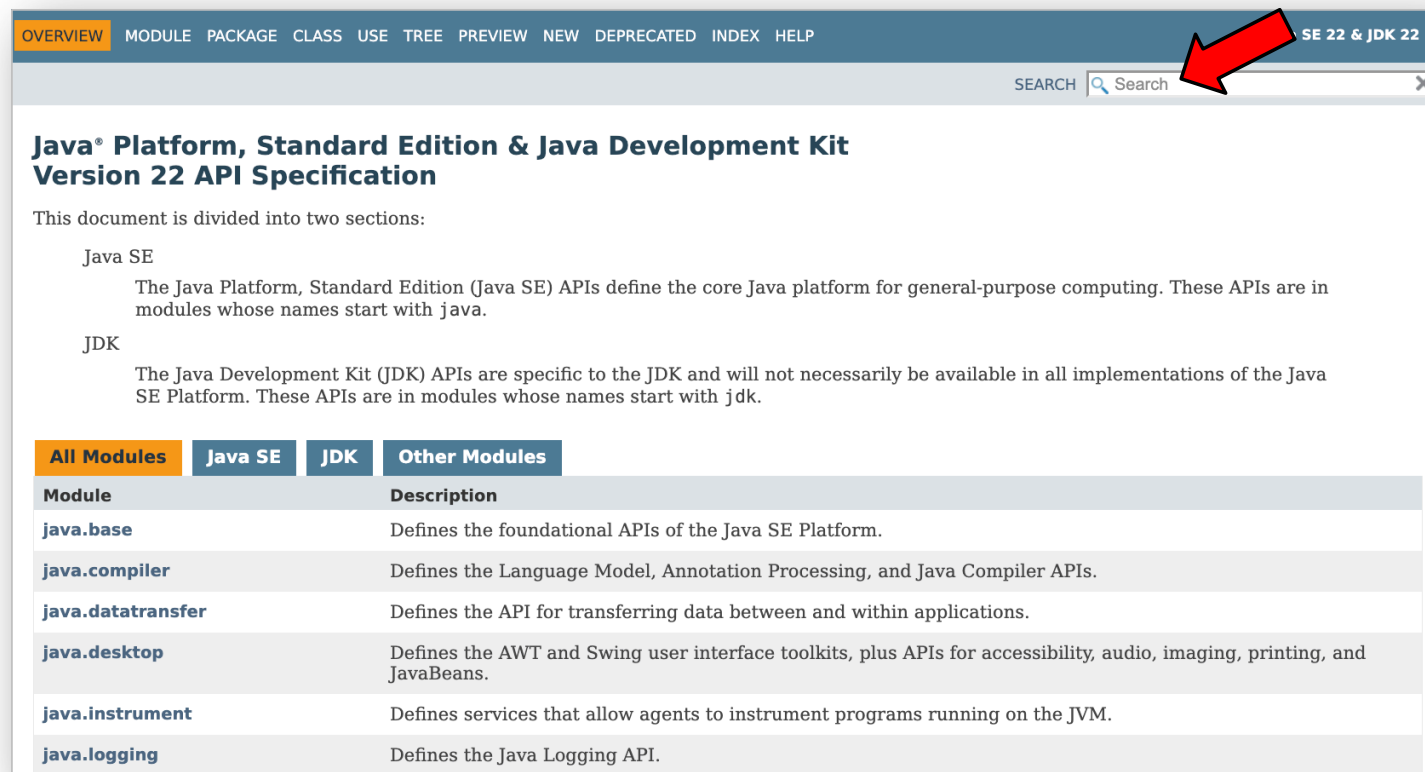
- Tools, Tutorials, API Documentation, ...

ex) <https://docs.oracle.com/en/java/javase/22>



# API Documentation (1/3)

- ex) <https://docs.oracle.com/en/java/javase/22/docs/api/index.html>
- Try Search 'System' for the reference of the class the class 'System'



The screenshot shows the Oracle Java SE 22 API Specification page. At the top, there is a navigation bar with links: OVERVIEW, MODULE, PACKAGE, CLASS, USE, TREE, PREVIEW, NEW, DEPRECATED, INDEX, and HELP. The version 'SE 22 & JDK 22' is displayed on the right. A search bar is located in the top right corner, with a red arrow pointing to it. The main heading is 'Java® Platform, Standard Edition & Java Development Kit Version 22 API Specification'. Below this, it states: 'This document is divided into two sections: Java SE' and 'JDK'. The Java SE section describes the core Java platform APIs, and the JDK section describes APIs specific to the JDK. At the bottom, there is a table with tabs for 'All Modules', 'Java SE', 'JDK', and 'Other Modules'. The 'All Modules' tab is selected, showing a table with columns 'Module' and 'Description'.

| Module                            | Description  |
|-----------------------------------|--|
| <a href="#">java.base</a>         | Defines the foundational APIs of the Java SE Platform.   |
| <a href="#">java.compiler</a>     | Defines the Language Model, Annotation Processing, and Java Compiler APIs.   |
| <a href="#">java.datatransfer</a> | Defines the API for transferring data between and within applications.   |
| <a href="#">java.desktop</a>      | Defines the AWT and Swing user interface toolkits, plus APIs for accessibility, audio, imaging, printing, and JavaBeans. |
| <a href="#">java.instrument</a>   | Defines services that allow agents to instrument programs running on the JVM.  |
| <a href="#">java.logging</a>      | Defines the Java Logging API.  |



# API Documentation (2/3)

The screenshot shows the Java API documentation for the `System` class. Red annotations highlight specific parts: two arrows point to the package information (`java.lang`) and the class hierarchy (`java.lang.Object` and `java.lang.System`), with the text "package including the class" next to them. Another arrow points to the "Field Summary" section, with the text "Fields: variables" next to it. The "Field Summary" section contains a table of static final fields.

OVERVIEW MODULE PACKAGE **CLASS** USE TREE PREVIEW NEW DEPRECATED INDEX HELP

SUMMARY: NESTED | FIELD | CONSTR | METHOD    DETAIL: FIELD | CONSTR | METHOD

Module `java.base`  
Package `java.lang`

**Class System**  
`java.lang.Object`  
`java.lang.System`

`public final class System`  
`extends Object`

The `System` class contains several useful class fields and methods. It cannot be instantiated. Among the facilities provided are: a means of accessing the underlying operating system; access to externally defined properties and environment variables; a means of loading files and libraries; and a utility for exiting the application.

Since:  
1.0

**Field Summary**

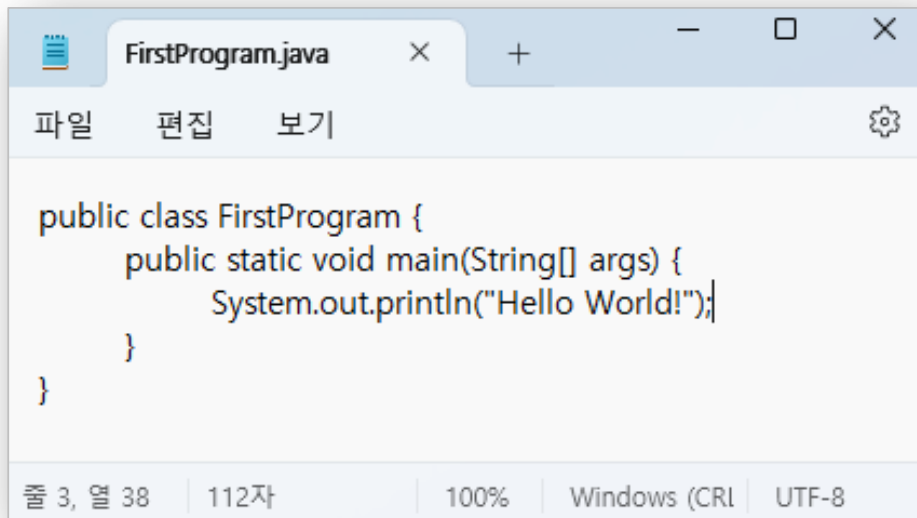
| Modifier and Type                     | Field            | Description                         |
|---------------------------------------|------------------|-------------------------------------|
| <code>static final PrintStream</code> | <code>err</code> | The "standard" error output stream. |
| <code>static final InputStream</code> | <code>in</code>  | The "standard" input stream.        |
| <code>static final PrintStream</code> | <code>out</code> | The "standard" output stream.       |

# API Documentation (3/3)

## Method Summary

| All Methods               | Static Methods   | Concrete Methods  | Deprecated Methods |
|---------------------------|--|---|--------------------|
| Modifier and Type         | Method   | Description   |                    |
| static void               | <code>arraycopy(Object src, int srcPos, Object dest, int destPos, int length)</code> | Copies an array from the specified source to the specified destination. |                    |
| static String             | <code>clearProperty(String key)</code>   | Removes the property with the specified key.                            |                    |
| static Console            | <code>console()</code>   | Returns the console, or null if none is available.                      |                    |
| static long               | <code>currentTimeMillis()</code>   | Returns the current time in milliseconds.                               |                    |
| static void               | <code>exit(int status)</code>  | Initiates the shutdown sequence with the specified status.              |                    |
| static void               | <code>gc()</code>  | Runs the garbage collection.  |                    |
| static Map<String,String> | <code>getenv()</code>  | Returns an unmodifiable map of environment variables.                   |                    |
| static String             | <code>getenv(String name)</code>   | Gets the value of the environment variable with the specified name.     |                    |
| static System.Logger      | <code>getLogger(String name)</code>  | Returns an instance of the logger with the specified name.              |                    |
| static System.Logger      | <code>getLogger(String name, ResourceBundle bundle)</code>                           | Returns a localized instance of the logger with the specified name.     |                    |

# Java Program Development Using JDK - 1

A screenshot of a text editor window titled "FirstProgram.java". The window has a menu bar with "파일" (File), "편집" (Edit), and "보기" (View), and a settings icon. The code inside is a simple Java class: 

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

 The status bar at the bottom shows "줄 3, 열 38" (Line 3, Column 38), "112자" (112 characters), "100%", "Windows (CRLF)", and "UTF-8".

- Type the program using Notepad (Windows) or Text editor (macOS)
- Save the program as "FirstProgram.java" (the same name as the class)

# Java Program Development Using JDK - 2

```
PS C:\Users\iklee\java> javac FirstProgram.java
PS C:\Users\iklee\java> dir

디렉터리: C:\Users\iklee\java

Mode                LastWriteTime         Length Name
----                -
-a----          2024-06-23 오후 11:26            430 FirstProgram.class
-a----          2024-06-23 오후 11:22            116 FirstProgram.java

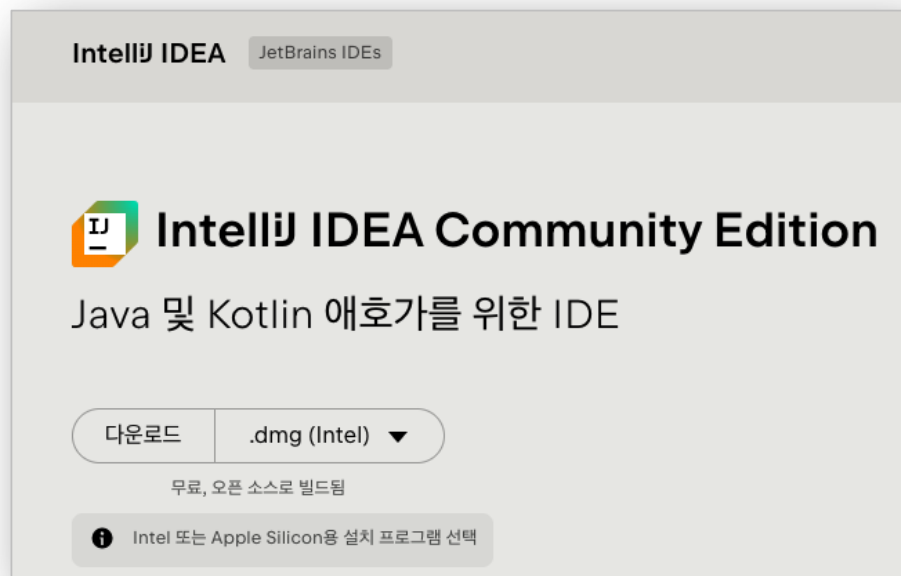
PS C:\Users\iklee\java> java FirstProgram
Hello World!
PS C:\Users\iklee\java>
```

"ls" command instead of "dir" in macOS

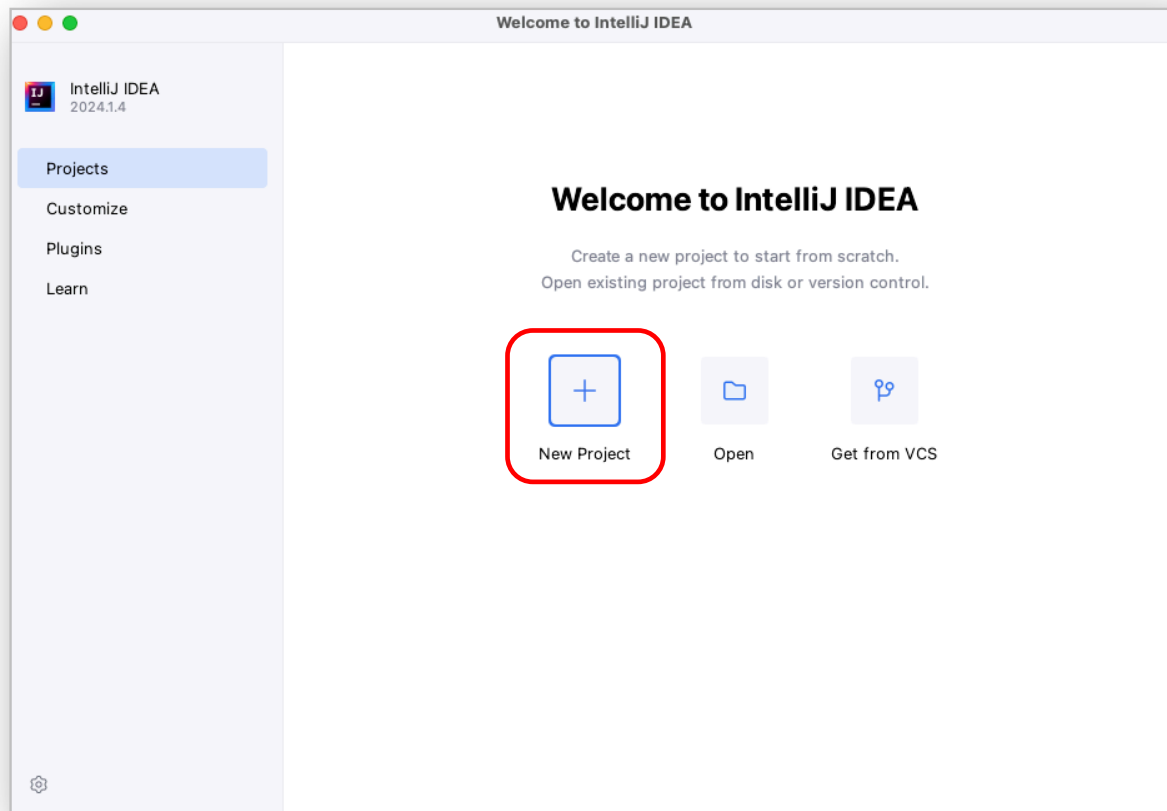
- Executing cmd (terminal in macOS)
- "javac FirstProgram.java" (compilation). Output is 'FirstProgram.class' (byte-code program)
- "java FirstProgram" (interpretation: executing the byte-code program on JVM)

# Installing IntelliJ IDEA

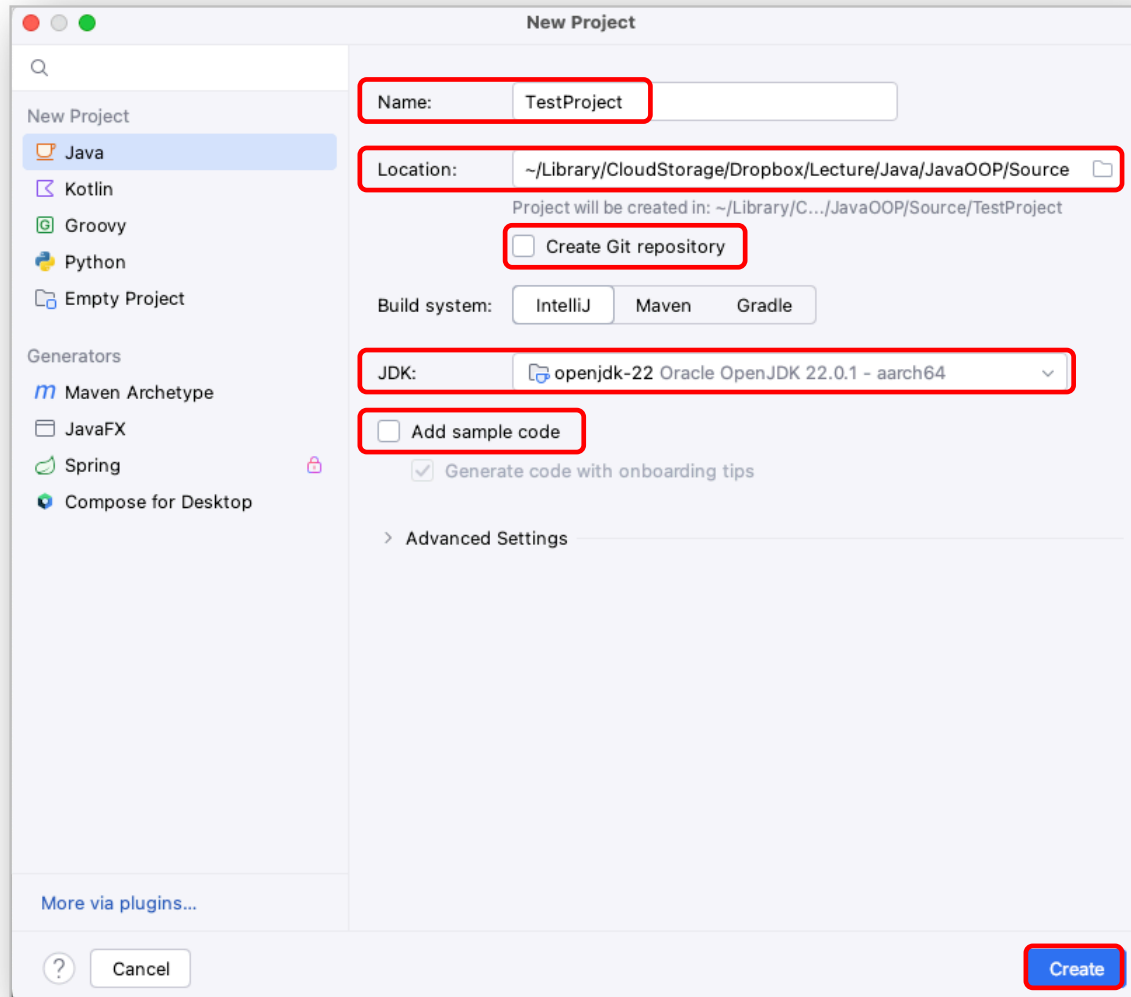
- <https://www.jetbrains.com/ko-kr/idea/download>
- Download “Community Edition” and Install it.
  - Windows, Intel mac, Apple Silicon mac



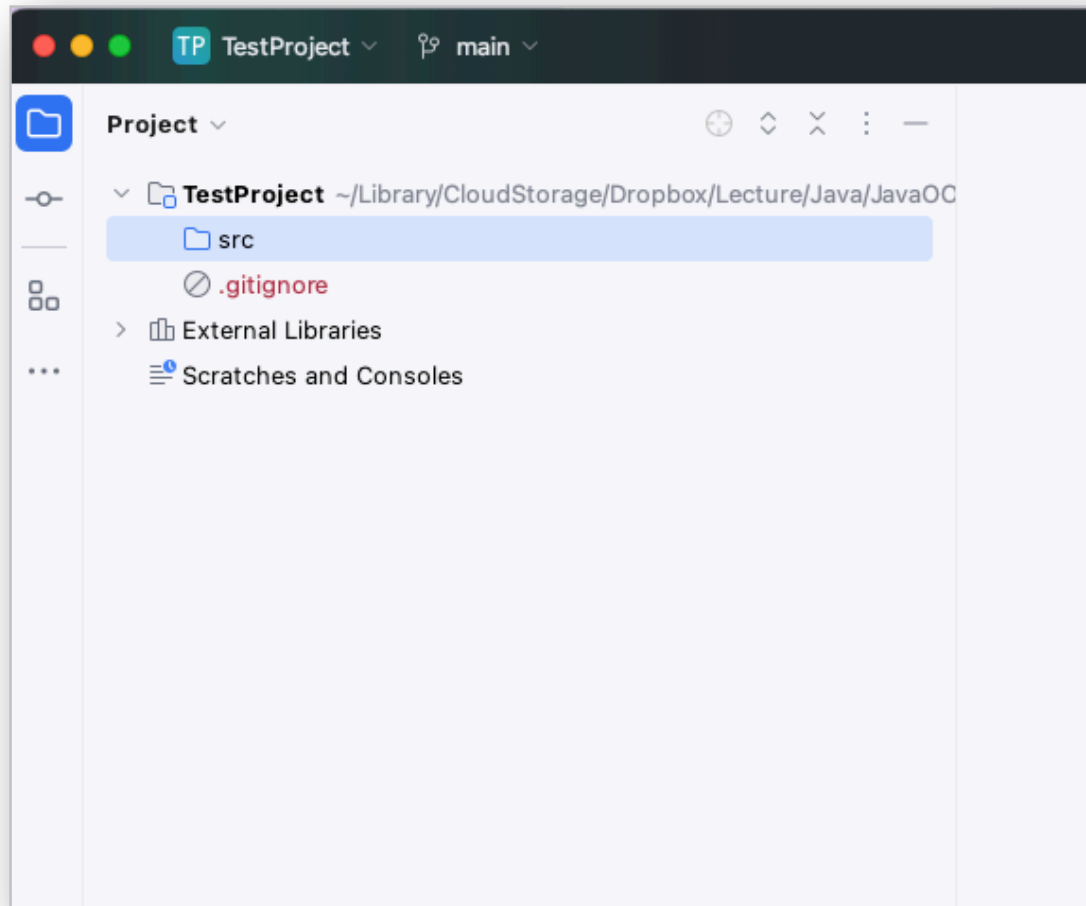
# Creating New Project (1/3)



# Creating New Project (2/3)

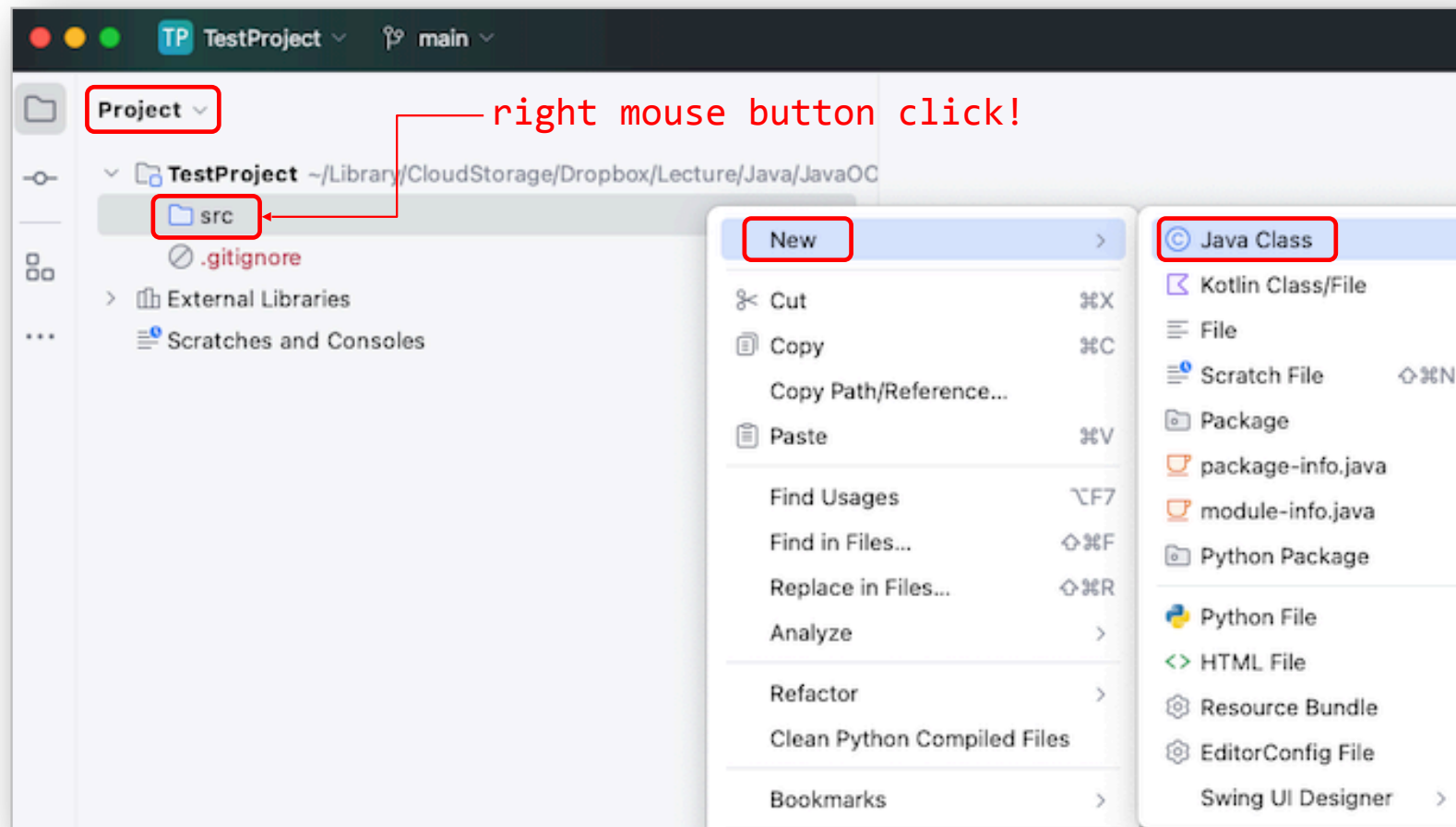


# Creating New Project (3/3)





# Creating New Class (1/3)



# Creating New Class (2/3)

New Java Class

TestClass

Class

Interface

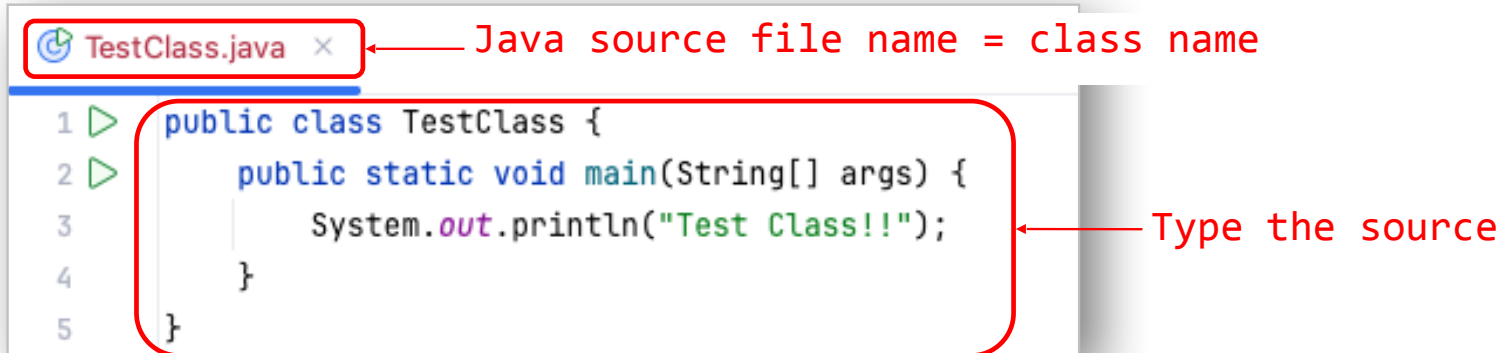
Record

Enum

Annotation

Type class name and enter!

# Creating New Class (3/3)



```
1 public class TestClass {  
2     public static void main(String[] args) {  
3         System.out.println("Test Class!!");  
4     }  
5 }
```

Java source file name = class name

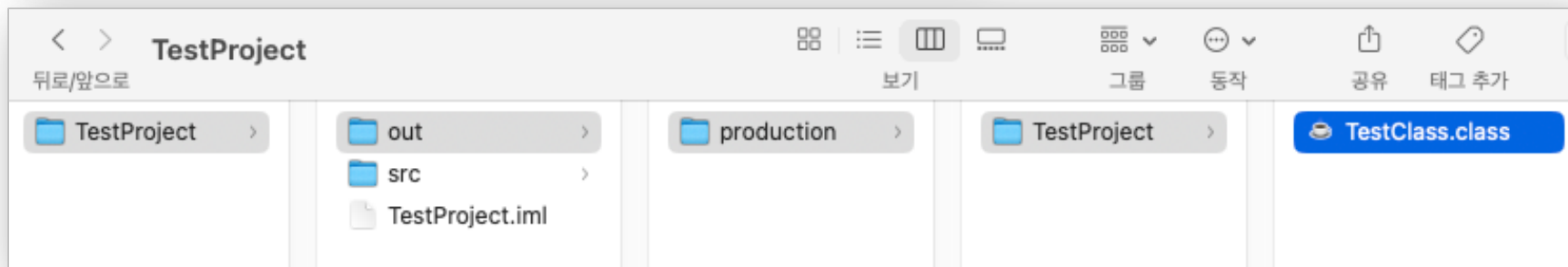
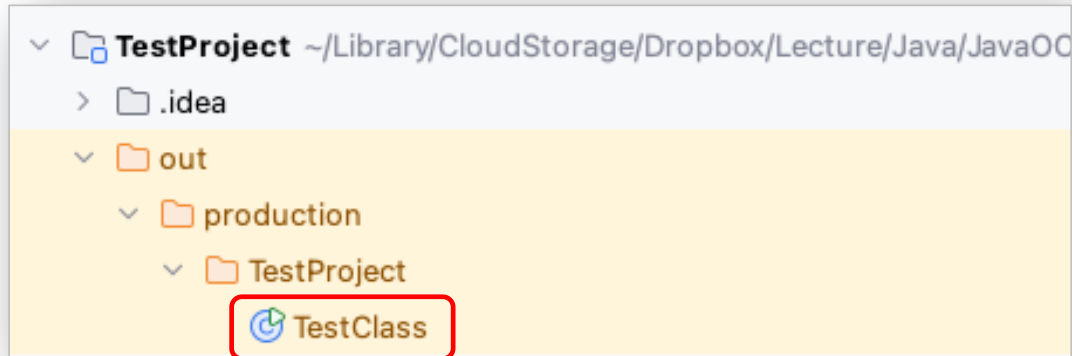
Type the source

# Build Project (Compilation)

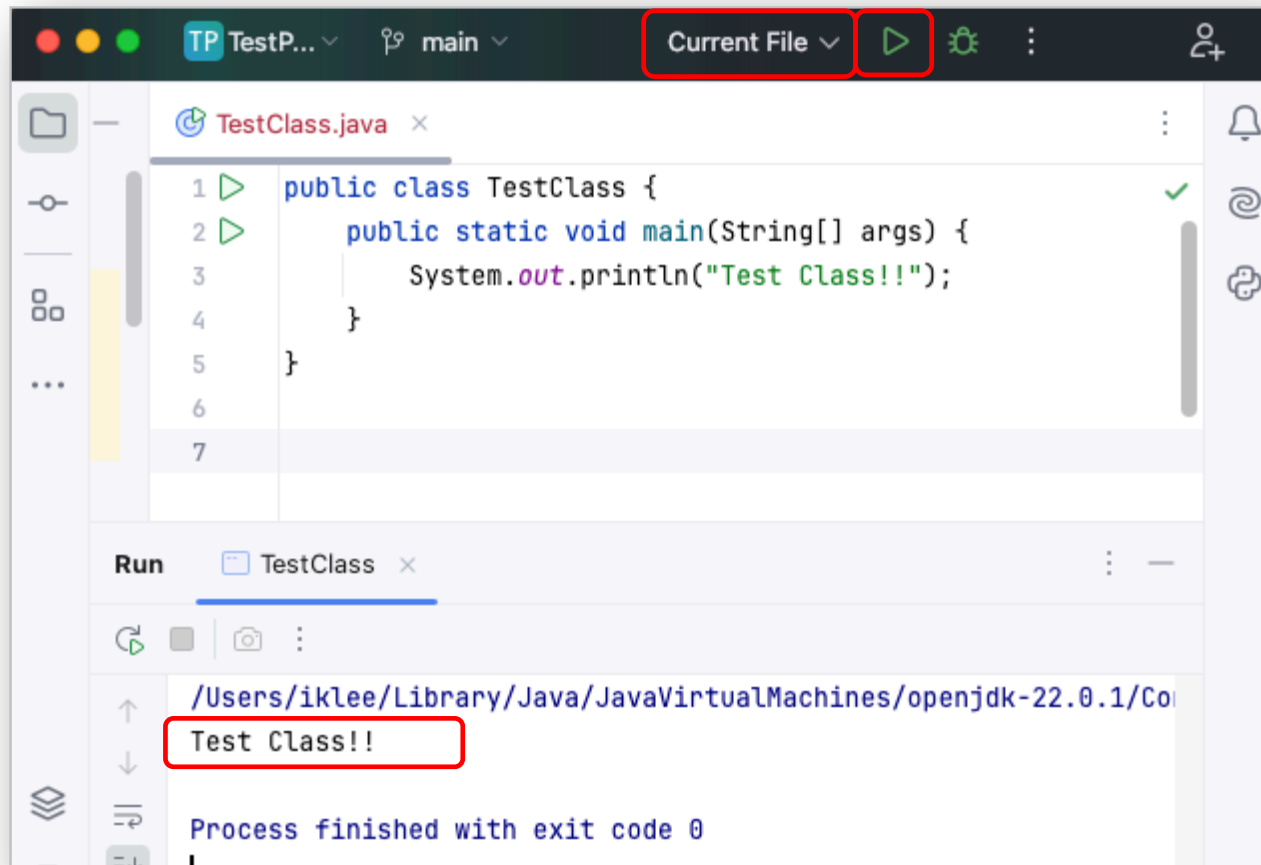
- Java Source to Bytecode
- Build Project
  - Build menu > Build Project
  - Windows
    - Ctrl + F9
  - MacOS
    - ⌘ fn F9 (Command + fn + F9)

# Bytecode

- Output of Compilation



# Run



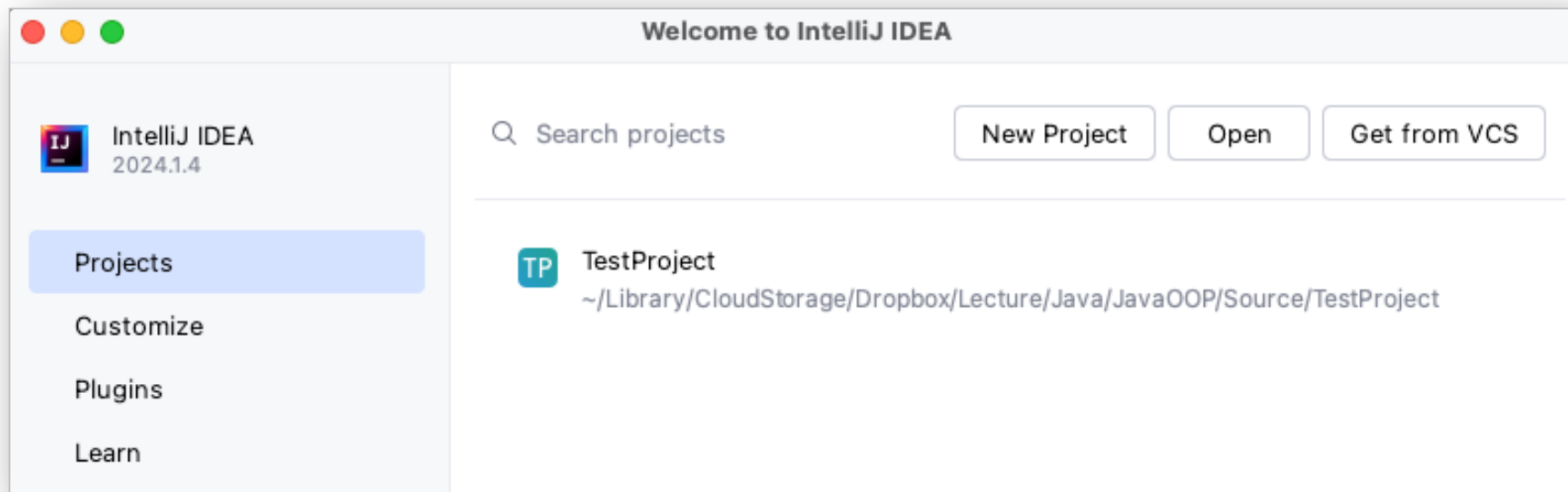
Ctrl + r for run  
(Interpreted by JVM)

or run button

Output of the program

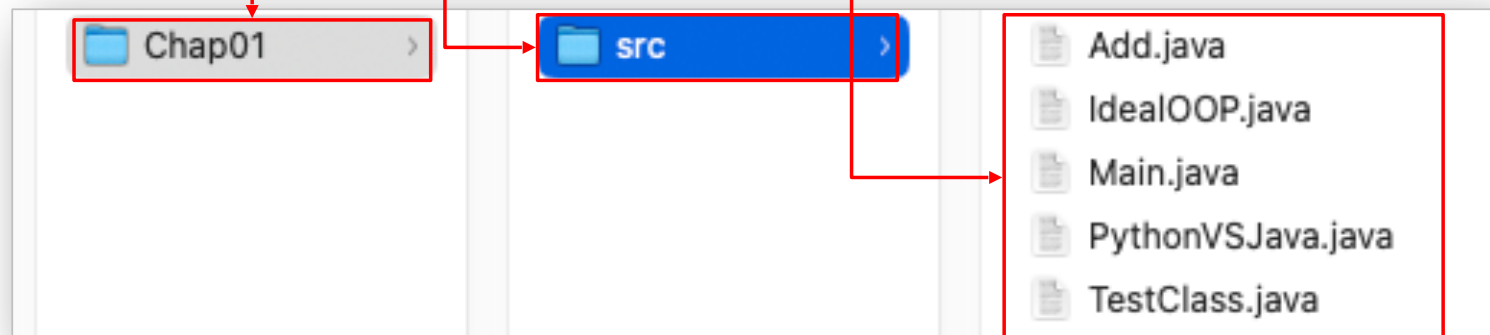
# Close Project

- Main menu "File > Close Project"
- After closing the project:



# New Project from Existing Source Files (1/3)

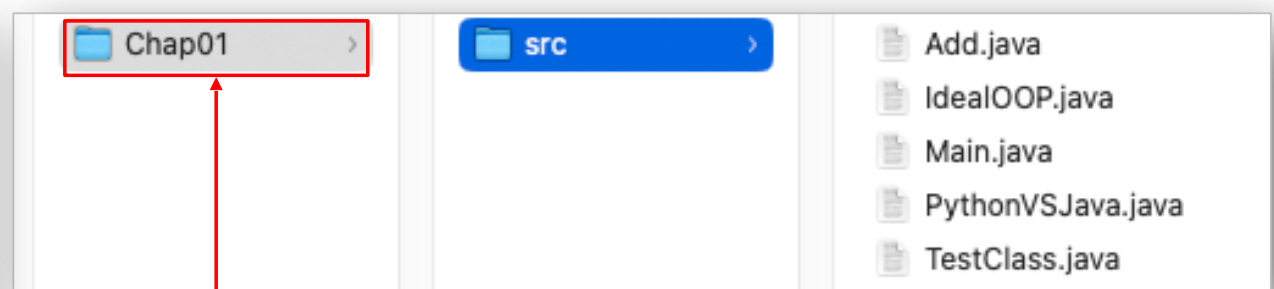
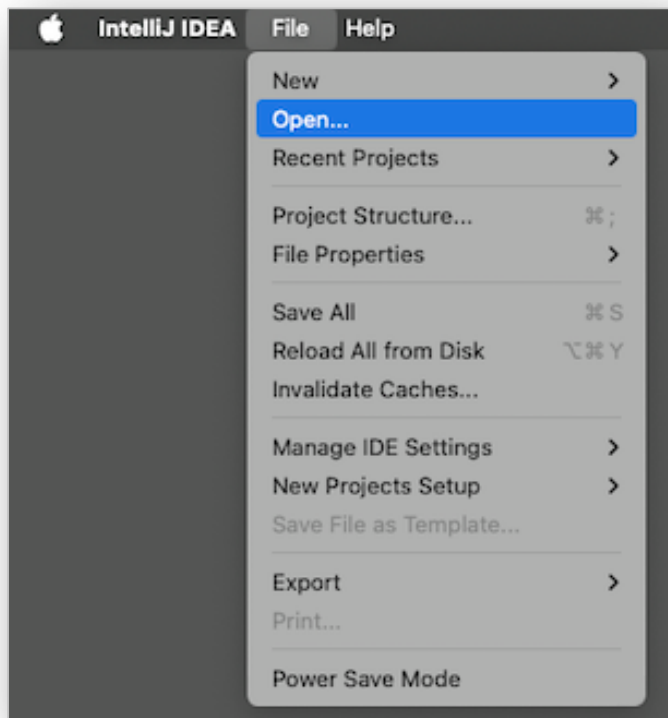
- Prepare the source
  - Project name > src > Java source files





# New Project from Existing Source Files (2/3)

- Main menu "File > Open"
- Select the project directory



Prepared Project Directory

# New Project from Existing Source Files (3/3)

