

ENG23 2032 – OBJECT ORIENTED TECHNOLOGY

Week 01 : Introduction to OOP



Ratiporn Chanklan

Assessment

10% Attendance

30% Weekly Lab Assignments:

10% Lab Tests:

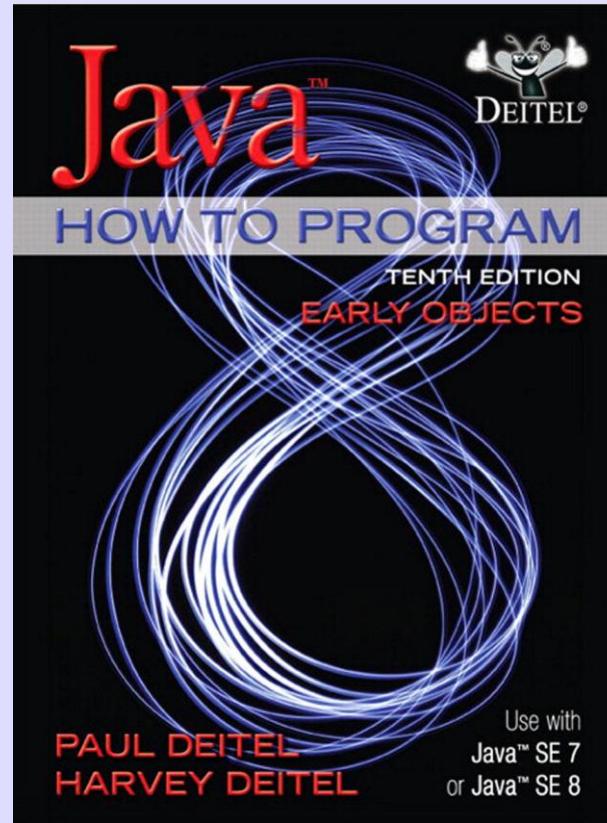
10% Lab Exam:

20% Midterm Exam:

20% Final Exam:



Indicative Reading

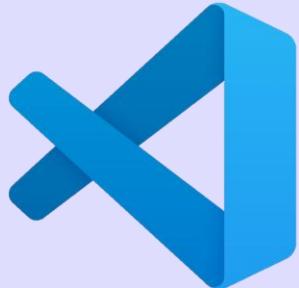


Paul Deitel and Harvey Deitel, *Java: How to Program (Early Objects) (10th Edition)*

What is Java?

- Java is a popular programming language, created in 1995.
- It is owned by Oracle, and more than 3 billion devices run Java.
- It is used for:
 - Mobile applications (specially Android apps)
 - Desktop applications
 - Web applications
 - Web servers and application servers
 - Games
 - Database connection
 - And much, much more!

Integrated Development Environment (IDE)



VS CODE

Visual Studio Code is a source code editor.

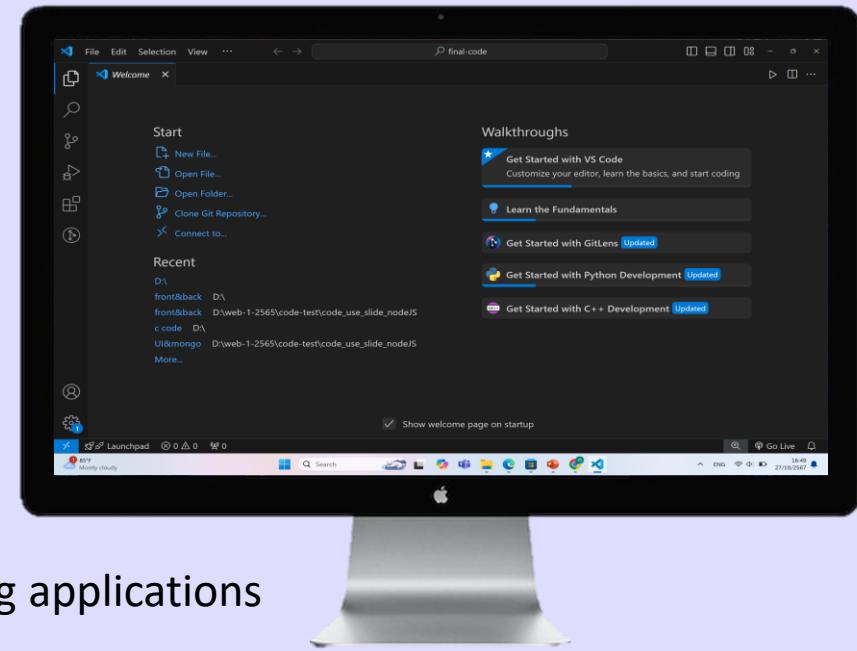
<https://code.visualstudio.com/>



JDK (Java Development Kit)

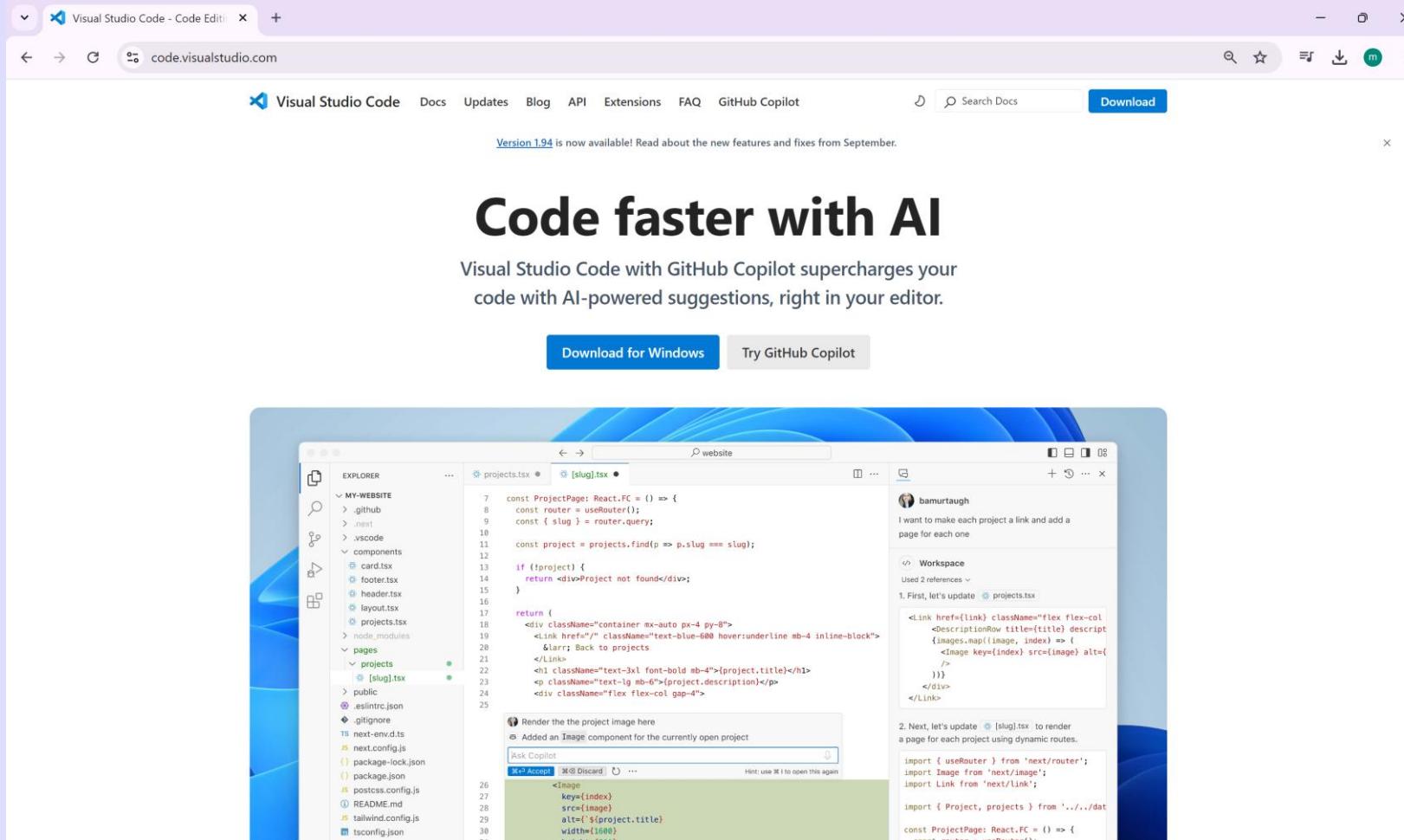
The JDK is a development environment for building applications using the Java programming language.

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



Install Visual Studio Code.

- VSCode Link:- <https://code.visualstudio.com/>



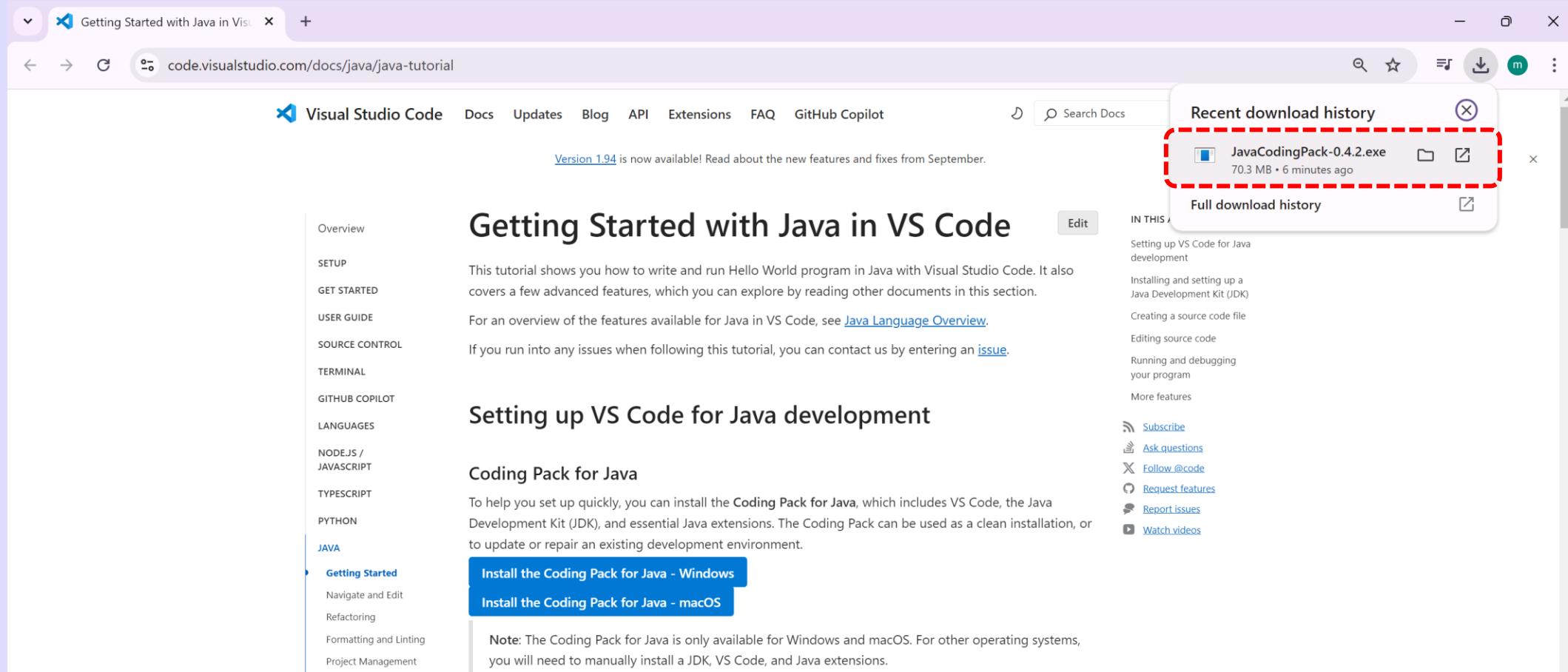
Install the Coding Pack for Java (1)

- <https://code.visualstudio.com/docs/java/java-tutorial>

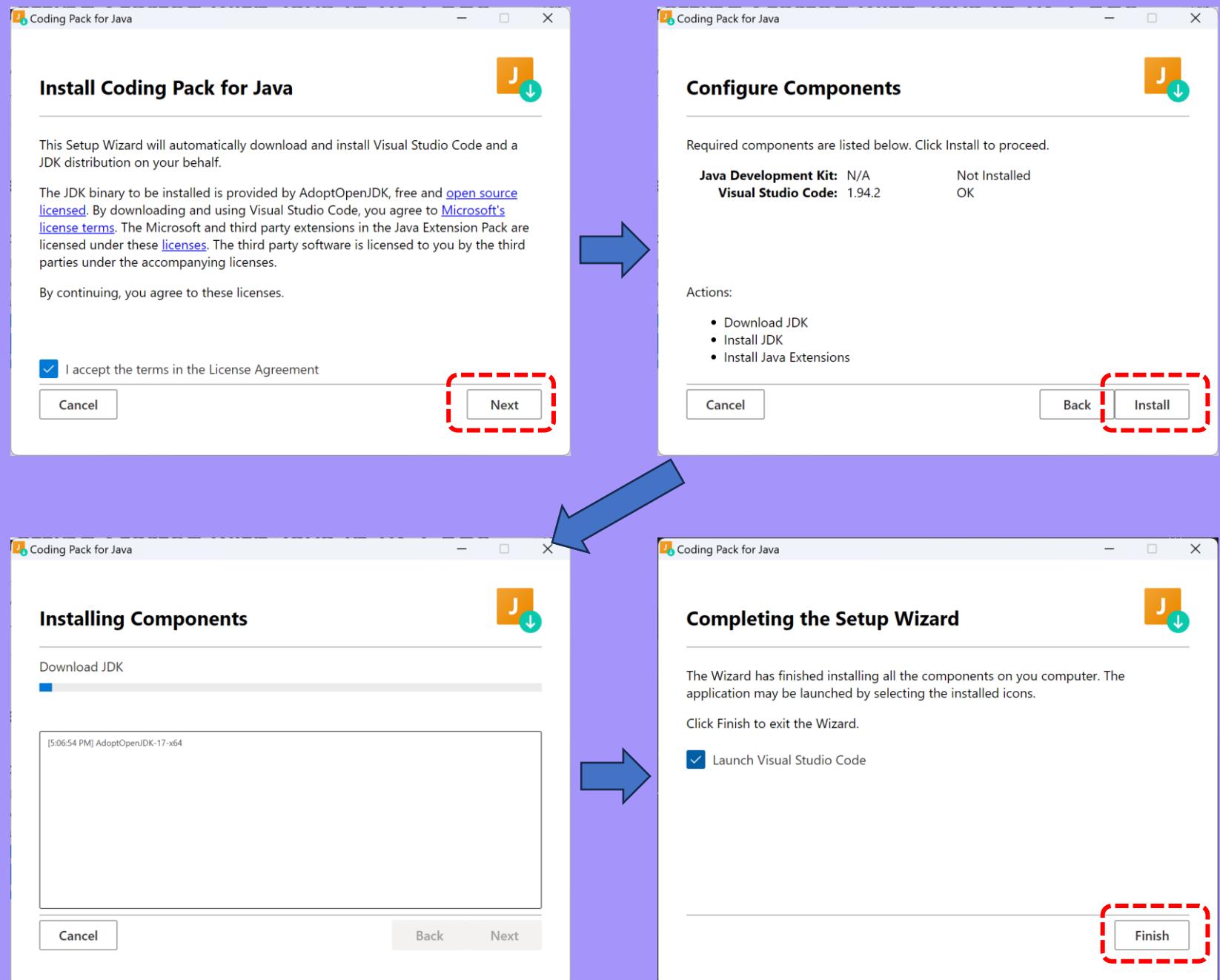
The screenshot shows a web browser displaying the Visual Studio Code Java documentation at code.visualstudio.com/docs/java/java-tutorial. The page title is "Getting Started with Java in VS Code". On the left, there's a sidebar with links for Overview, SETUP, GET STARTED, USER GUIDE, SOURCE CONTROL, TERMINAL, GITHUB COPILOT, LANGUAGES, NODEJS / JAVASCRIPT, TYPESCRIPT, PYTHON, and JAVA. Under the JAVA section, "Getting Started" is selected. Below the sidebar, there's a note about Version 1.94 and a search bar. The main content area starts with a heading "Getting Started with Java in VS Code". It includes a paragraph about the tutorial, a link to the Java Language Overview, and a note about reporting issues. A large heading "Setting up VS Code for Java development" follows. Under this, there's a section for "Coding Pack for Java" with a note about its availability for Windows and macOS. Two blue buttons are shown: "Install the Coding Pack for Java - Windows" and "Install the Coding Pack for Java - macOS". A red dashed box highlights these two buttons. To the right of the main content, there's a sidebar titled "IN THIS ARTICLE" with links to Setting up VS Code for Java development, Installing and setting up a Java Development Kit (JDK), Creating a source code file, Editing source code, Running and debugging your program, and More features. At the bottom, there are links for Subscribe, Ask questions, Follow @code, Request features, Report issues, and Watch videos.

Install the Coding Pack for Java (2)

- Once you finish downloading the coding pack, click on the file to install it.



Install the Coding Pack for Java (3)



Check your the Coding Pack for Java installation.

- Click on a search type in CMD
- Open a new Command Prompt and type: **java --version**



A screenshot of a Windows Command Prompt window titled "Command Prompt". The window shows the following output:

```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Lenovo>java --version
openjdk 17.0.13 2024-10-15
OpenJDK Runtime Environment Temurin-17.0.13+11 (build 17.0.13+11)
OpenJDK 64-Bit Server VM Temurin-17.0.13+11 (build 17.0.13+11, mixed mode, sharing)

C:\Users\Lenovo>
```

Installing extensions (1)

- <https://code.visualstudio.com/docs/java/java-tutorial>

The screenshot shows a web browser window displaying the Visual Studio Code Java tutorial at code.visualstudio.com/docs/java/java-tutorial. The page has a light purple header bar with the URL. Below it is a standard browser navigation bar. The main content area has a white background. On the left, there's a sidebar with a tree view of topics under 'JAVA'. The 'Getting Started' section is expanded, showing 'Navigate and Edit', 'Refactoring', 'Formatting and Linting', 'Project Management', 'Build Tools', 'Run and Debug', and 'Testing'. A red dashed box highlights the 'Install the Extension Pack for Java' button, which is a blue call-to-action button with white text. To the right of the sidebar, the main content area starts with a note about manually installing a JDK, VS Code, and Java extensions. It then has a section titled 'Installing extensions' with a list of Java-related extensions. Below this is a large dark callout box titled 'Tips for Beginners' containing a 'Quick Start' guide. On the far right, there's a sidebar titled 'IN THIS ARTICLE' with links to other documentation sections like 'Setting up VS Code for Java development', 'Creating a source code file', etc., and social sharing links for 'Subscribe', 'Ask questions', etc.

you will need to manually install a JDK, VS Code, and Java extensions.

Installing extensions

If you are an existing VS Code user, you can also add Java support by installing the [Extension Pack for Java](#), which includes these extensions:

- [Language Support for Java™ by Red Hat](#)
- [Debugger for Java](#)
- [Test Runner for Java](#)
- [Maven for Java](#)
- [Project Manager for Java](#)
- [Visual Studio IntelliCode](#)

Install the Extension Pack for Java

The [Extension Pack for Java](#) provides a Quick Start guide and tips for code editing and debugging. It also has a FAQ that answers some frequently asked questions. Use the command **Java: Tips for Beginners** from the Command Palette (**Ctrl+Shift+P**) to launch the guide.

Tips for Beginners

Quick Start Code Editing Debugging FAQ

In this 1-minute tutorial, we'll show you how to create a quick-start Java program in VS Code.

Setup the Workspace

VS Code Java works directly with **folders** that have source code. To setup the workspace, simply open a folder using File > Open Folder...

IN THIS ARTICLE

- { [Setting up VS Code for Java development](#)
- Installing and setting up a Java Development Kit (JDK)
- Creating a source code file
- Editing source code
- Running and debugging your program
- More features

[Subscribe](#) [Ask questions](#) [Follow @code](#) [Request features](#) [Report issues](#) [Watch videos](#)

Installing extensions (2)

A screenshot of a web browser window titled "Getting Started with Java in Visual Studio Code". The URL is code.visualstudio.com/docs/java/java-tutorial. A modal dialog box is centered over the page, asking "Open Visual Studio Code?". It contains the text: "https://code.visualstudio.com wants to open this application." and a checkbox "Always allow code.visualstudio.com to open links of this type...". Below the checkbox is a list of Java extensions: Language Support for Java, Debugger for Java, Test Runner for Java, Maven for Java, Project Manager for Java, and Visual Studio IntelliCode. At the bottom of the modal is a red dashed box highlighting the "Open Visual Studio Code" button. To the right of the modal, the main page content is visible, including a "Search Docs" bar, a "Download" button, and a sidebar titled "IN THIS ARTICLE" with links like "Setting up VS Code for Java development", "Editing source code", and "More features".

Getting Started with Java in Visual Studio Code

code.visualstudio.com/docs/java/java-tutorial

Open Visual Studio Code?

https://code.visualstudio.com wants to open this application.

Always allow code.visualstudio.com to open links of this type...

Open Visual Studio Code Cancel

IN THIS ARTICLE

Setting up VS Code for Java development

Editing source code

More features

Subscribe Ask questions Follow @code Request features Report issues Watch videos

Overview

SETUP

GET STARTED

USER GUIDE

SOURCE CONTROL

TERMINAL

GITHUB COPILOT

LANGUAGES

NODE.JS / JAVASCRIPT

TYPESCRIPT

PYTHON

JAVA

Getting Started

Navigate and Edit

Refactoring

Formatting and Linting

Project Management

Build Tools

Run and Debug

Testing

Spring Boot

Language Support for Java

Debugger for Java

Test Runner for Java

Maven for Java

Project Manager for Java

Visual Studio IntelliCode

Install the Extension Pack for Java

The Extension Pack for Java provides a Quick Start guide and tips for code editing and debugging. It also has a FAQ that answers some frequently asked questions. Use the command Java: Tips for Beginners from the Command Palette (`Ctrl+Shift+P`) to launch the guide.

Tips for Beginners

Quick Start Code Editing Debugging FAQ

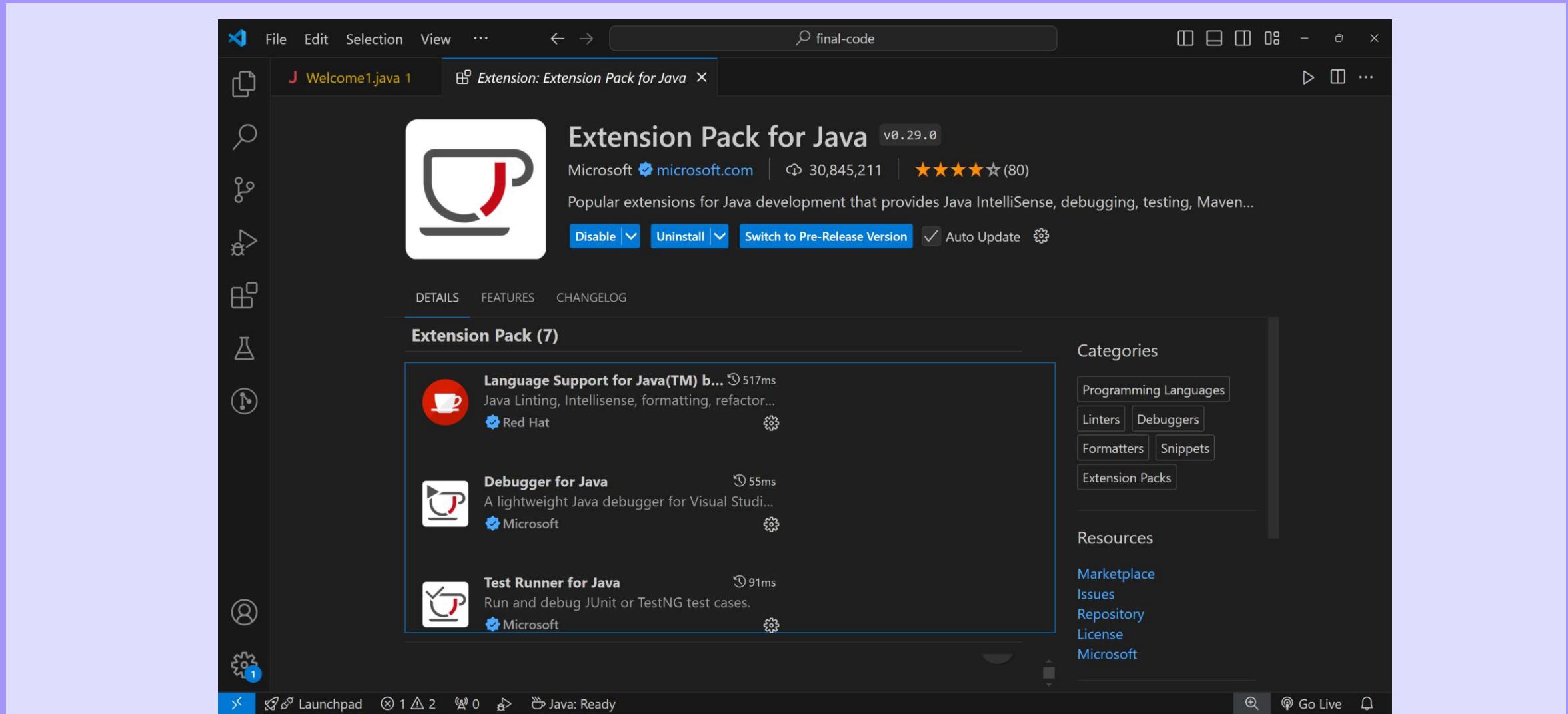
In this 1-minute tutorial, we'll show you how to create a quick-start Java program in VS Code.

Setup the Workspace

VS Code Java works directly with **folders** that have source code. To setup the workspace, simply open a folder using File > Open Folder...

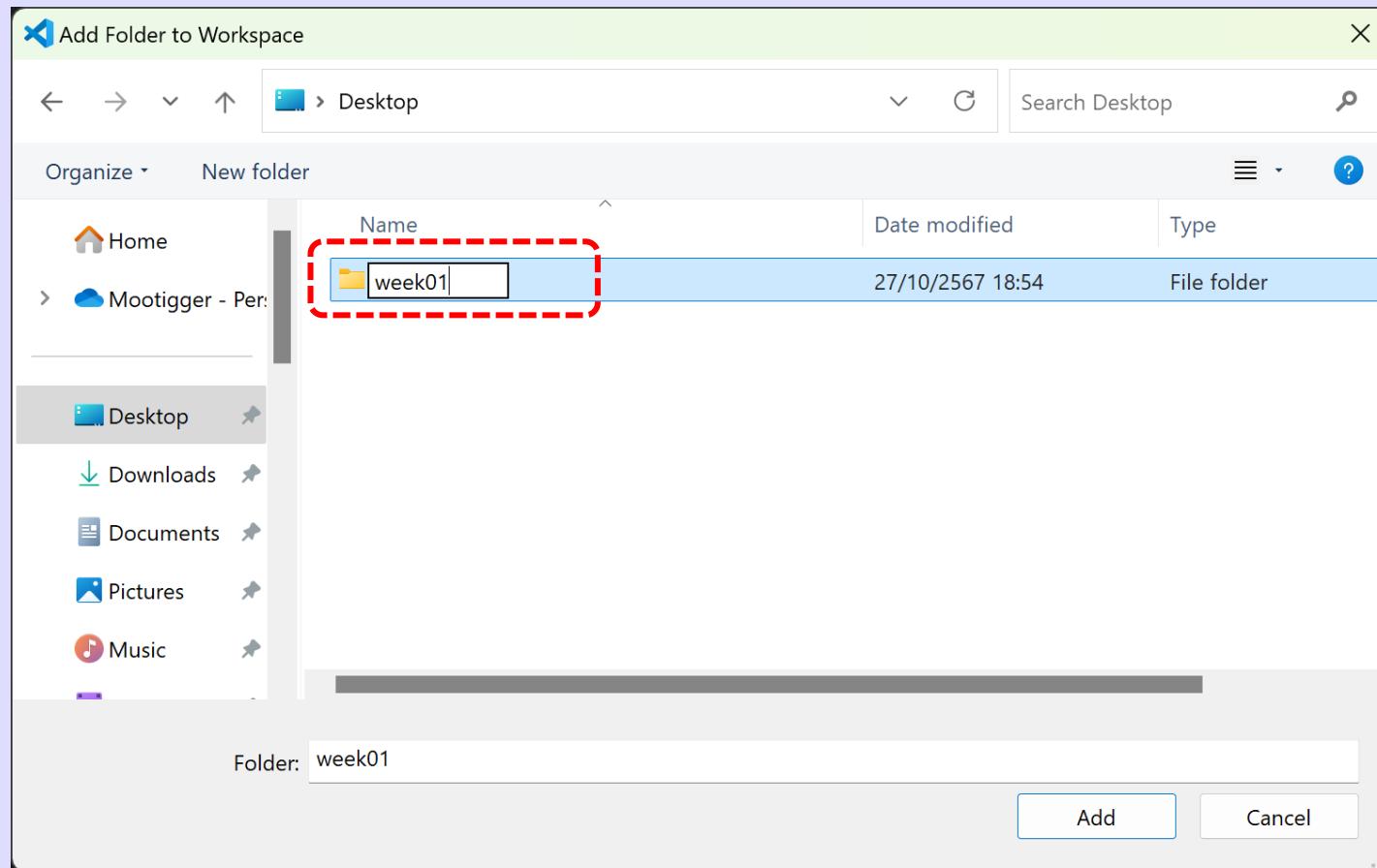
Create a Class

Installing extensions (3)



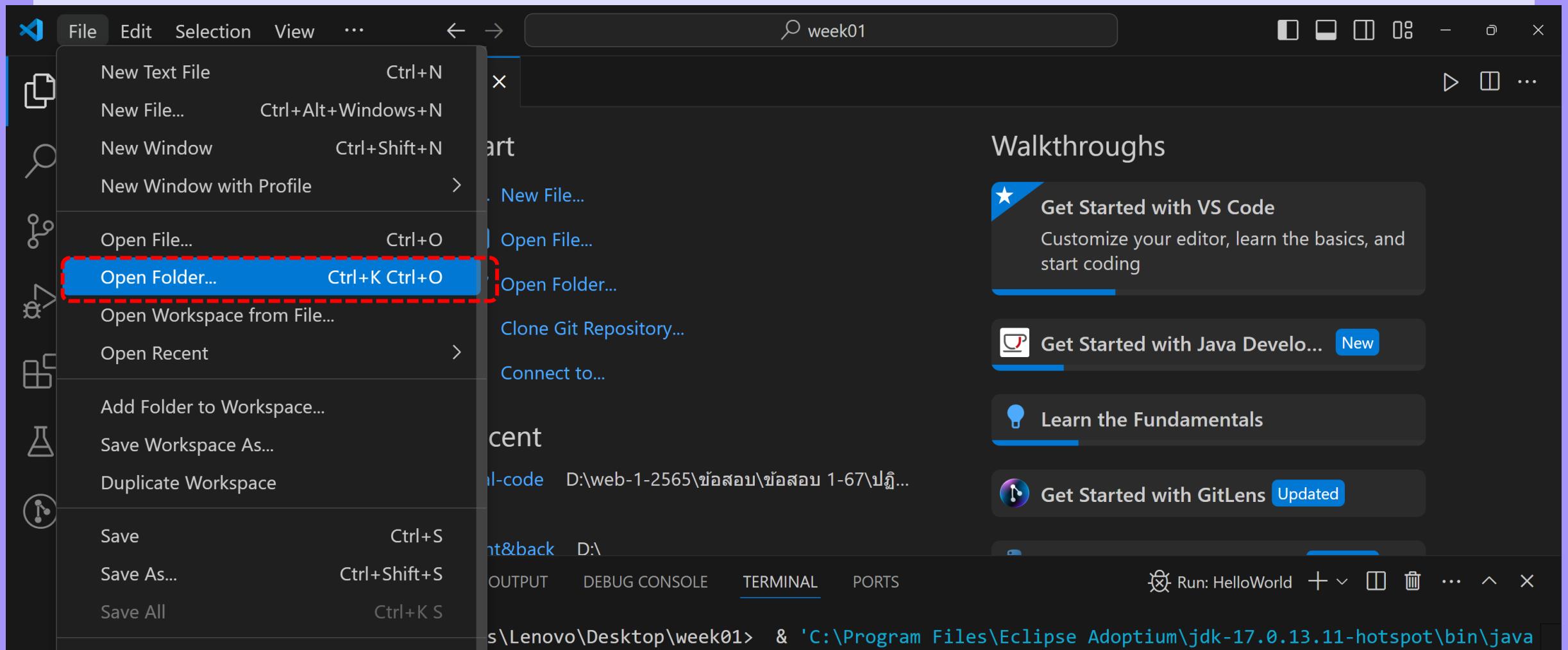
Creating a source code file (1)

Create a folder to save .java files.



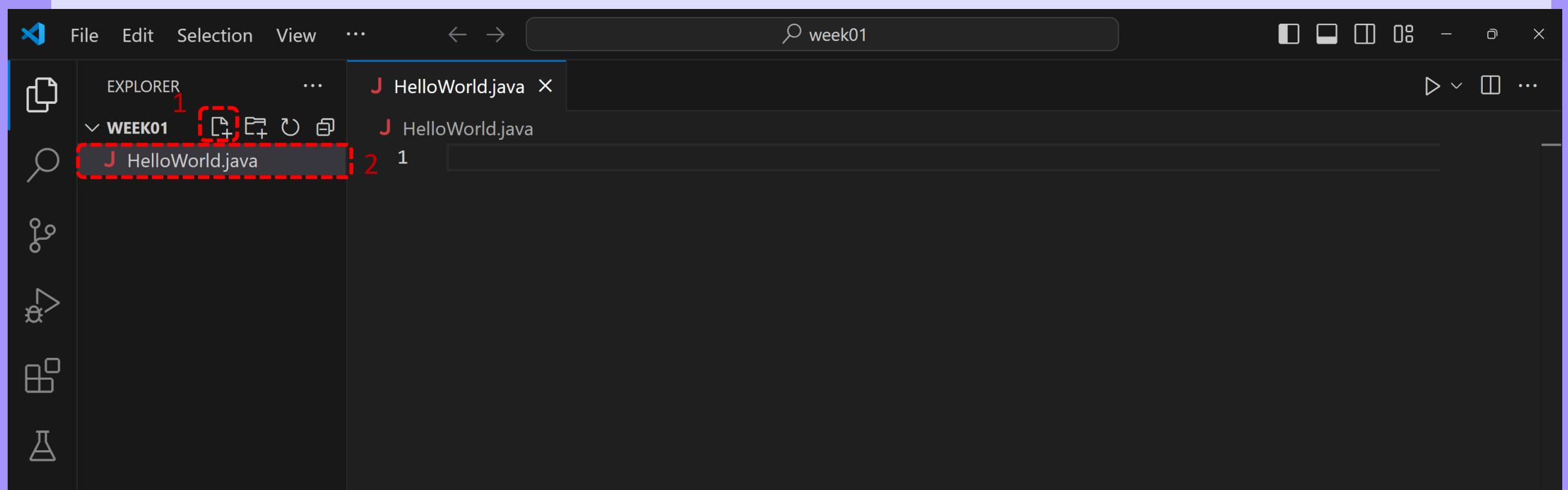
Creating a source code file (1)

- File->Open Folder...->select the folder you created before (week01)



Creating a source code file (3)

- create a new file and save it with the name HelloWorld.java.



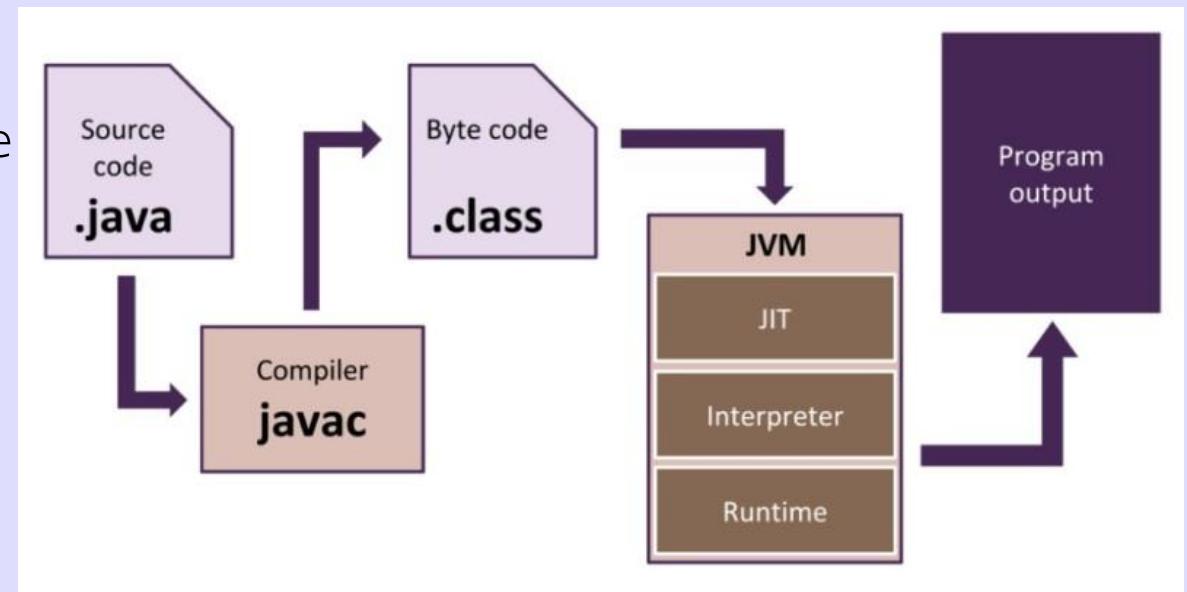
Creating a source code file (4)

- Input a source code in HelloWorld.java.

```
public class HelloWorld {  
    public static void main(String[] args){  
        System.out.println("Welcome to Java Programming!");  
    }  
}
```

Run .java

- type "**javac filename.java**": This will compile your code. If there are no errors in the code, the command prompt will take you to the next line.
- If there are no errors, It converting the code written in **filename.java** into bytecode (a **filename.class** file will be generated), which the JVM (Java Virtual Machine) can execute.
- The bytecode is saved in a **.class** file.
- Now, type "**java filename.java**" to run the file



Run .java

The screenshot shows a Java development environment with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Search Bar:** week01
- Toolbar:** Minimize, Maximize, Close
- Left Sidebar (EXPLORER):** WEEK01 folder containing HelloWorld.class and HelloWorld.java (selected).
- Code Editor:** HelloWorld.java content:

```
public class HelloWorld {  
    public static void main(String[] args){  
        System.out.println("Welcome to Java Programming!");  
    }  
}
```
- Tool Buttons:** Run | Debug
- Bottom Navigation:** PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS
- Terminal:** PowerShell session showing:

```
PS C:\Users\Lenovo\Desktop\week01> javac HelloWorld.java  
PS C:\Users\Lenovo\Desktop\week01> java HelloWorld.java  
Welcome to Java Programming!  
PS C:\Users\Lenovo\Desktop\week01>
```

The command "javac HelloWorld.java" and the resulting output "Welcome to Java Programming!" are highlighted with a red dashed box.
- Bottom Right:** PowerShell icon (x2)

Run .java

The screenshot shows a Java code editor interface with the following details:

- File Menu:** File, Edit, Selection, View, ...
- Search Bar:** week01
- Toolbar:** Minimize, Maximize, Close
- Explorer:** WEEK01, HelloWorld.java
- Code Editor:** HelloWorld.java file open, containing the following code:

```
1 public class HelloWorld {  
2     public static void main(String[] args){  
3         System.out.println("Welcome to Java Programming!");  
4     }  
5 }
```

A red arrow points from the bottom left towards the code editor area.
- Context Menu:** Run, Debug (highlighted with a red box)
- Terminal:** Shows the command run and the output:

```
PS C:\Users\Lenovo\Desktop\week01> & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.13.11-hotspot\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\Lenovo\AppData\Roaming\Code\User\workspaceStorage\66b5db44a151f561ffff1877a78830c6d\redhat.java\jdt_ws\week01_f6e31805\bin' 'HelloWorld'  
Welcome to Java Programming!  
PS C:\Users\Lenovo\Desktop\week01>
```

A red dashed box highlights the terminal output area.

Declaring a Class & Class Body

Class Declaration

```
public class HelloWorld{
    public static void main(String[] args){
        System.out.println("Welcome to Java Programming!");
    }
}
```

Class Body

- **public class HelloWorld** begins a class declaration for class HelloWorld.
- Every Java program consists of at least one class that you (the programmer) define.
- The **class** keyword introduces a class declaration and is immediately followed by the class name (HelloWorld).
- A **public class** must be placed in a file that has a filename of the form **ClassName.java**, so **class HelloWorld** is stored in the file **HelloWorld.java**
- **Class names begin** with a **capital letter** and **capitalize the first letter of each word** they include (e.g., SampleClassName).
- A class name is an identifier—a series of characters consisting of letters, digits, underscores (_) and dollar signs (\$) that does not begin with a digit and does not contain spaces.
- **Class Body**, A left brace, {, begins the **body** of every **class declaration**. A corresponding right brace, }, must end each class declaration.

Declaring a Method

```
public class HelloWorld {    Method Declaration
    public static void main(String[] args){
        System.out.println("Welcome to Java Programming!");
    }
}
```

The code snippet shows a Java class declaration for "HelloWorld". The "main" method is highlighted with a yellow dashed box and labeled "Method Declaration". The code within the method, including the println statement, is highlighted with an orange dashed box and labeled "method body".

- `public static void main(String[] args)` is the starting point of every Java application.
- The **parentheses** after the identifier `main` indicate that it's a program building block called a method.
- Java class declarations normally contain one or more methods.
- Keyword `void` indicates that this **method will not return** any information.
- The `String[] args` in parentheses is a **required part of the method main's declaration**
- The left brace begins the body of the method declaration. A corresponding right brace must end it.

Performing Output with `System.out.println`

```
public class HelloWorld {  
    public static void main(String[] args){  
        System.out.println("Welcome to Java Programming!");  
    }  
}
```

- Method `System.out.print` displays (or prints) a line of text in the command window.
- Method `System.out.println` displays (or prints) a line of text by a new line in the command window.
- The string in the parentheses "Welcome to Java Programming!" is the argument to the method.
- including `System.out.println`, the argument "Welcome to Java Programming!" in the parentheses and the semicolon (;), is called a statement.
- A method typically contains one or more statements that perform its task.
- Most statements end with a semicolon.

Escape Sequences

Sequence	Description	Example	Output
\n	New Line	System.out.println("Hello\nWorld");	Hello World
\t	Tab	System.out.println("Name:\tJohn");	Name: John
\"	Double Quote	System.out.println("She said \"Hi\".");	She said "Hi".
'	Single Quote	System.out.println('\A');	'A'
\\\	Backslash	System.out.println("Path: C:\\dir");	Path: C:\dir
\r	Carriage Return	System.out.print("123\rABC");	ABC
\b	Backspace	System.out.print("Test\bA");	TesA

Exercise

- Create Welcome.java
- Write a Java program for Printing a line of text with multiple statements.
- Display the output of the program as follows:

```
Welcome to  
ENG23 2032 - OBJECT ORIENTED TECHNOLOGY
```

- Compile&Run
 - javac -d . week01/lecture/Welcome.java
 - java week01.lecture.Welcome