Core Java

Lesson 02: Getting Started

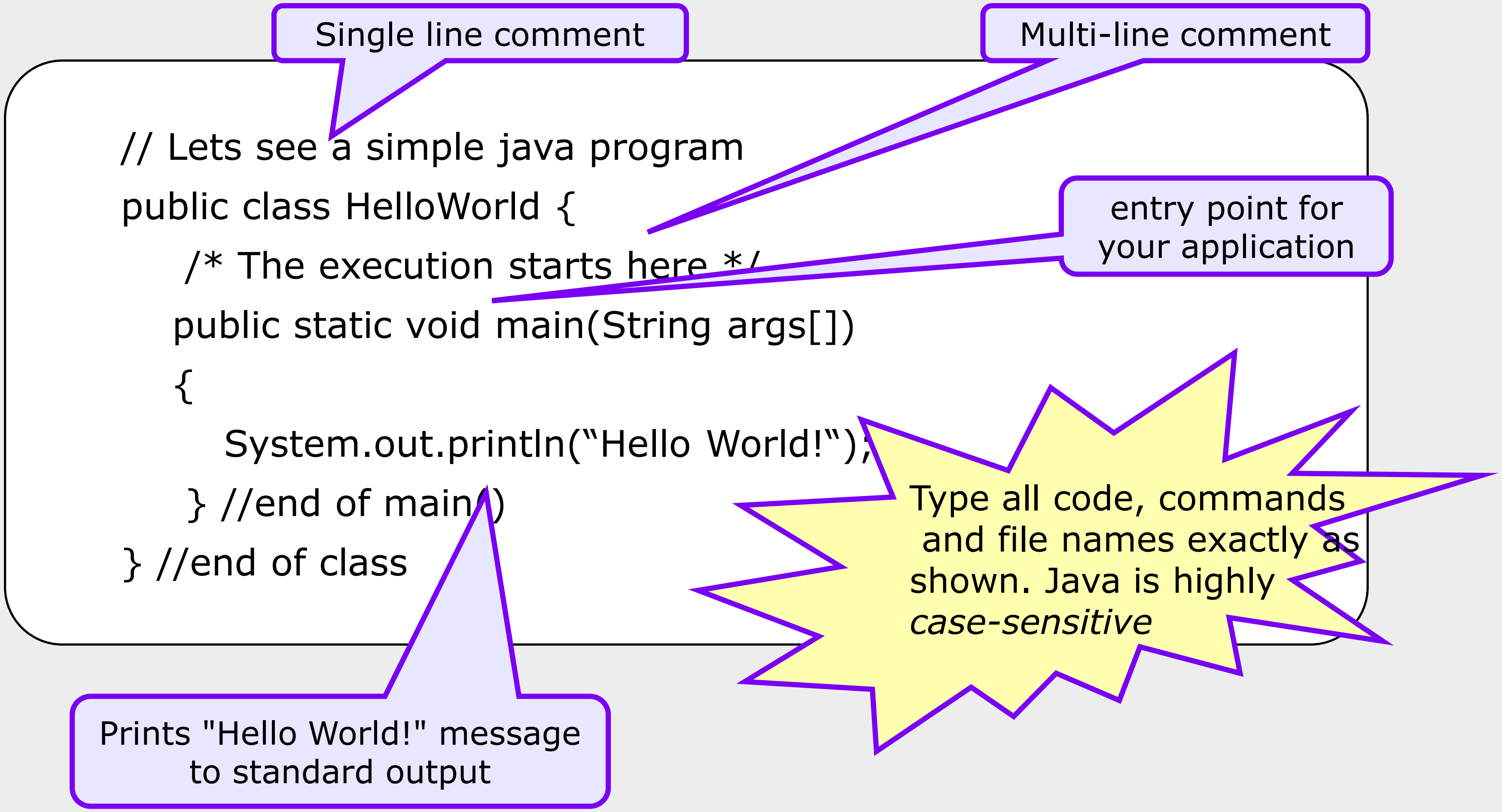
Lesson Objectives

- In this lesson you will learn -
  - Introduction to Java
  - Platform Independency in Java
  - Integrated Development Environment
  - Some Important Terms in Java
  - JVM Basic Architecture

What is Java?

- Java is an Object-Oriented programming language – most of it is free and open source!
  - It is developed in the early 1990s, by James Gosling of Sun Microsystems
  - It allows development of software applications.
  - It is amongst the preferred choice for developing internet-based applications

A Sample Program

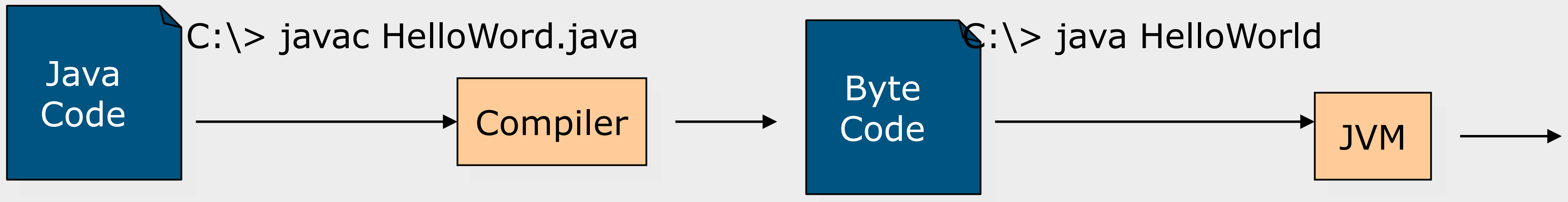


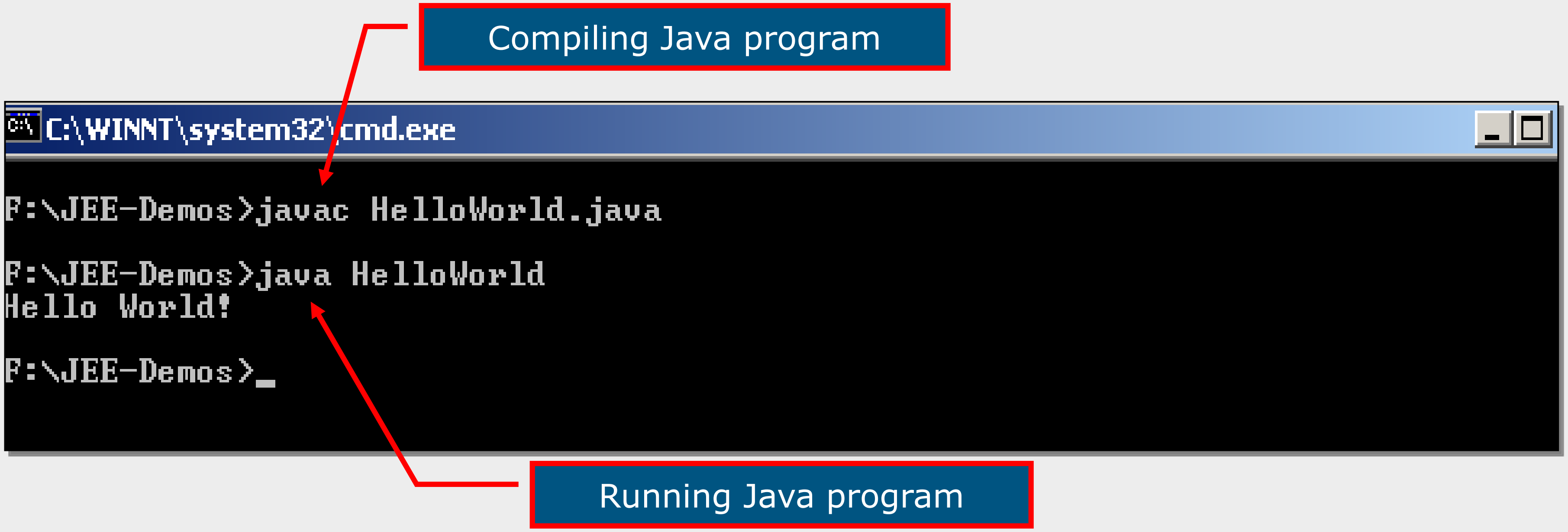
```
// Lets see a simple java program
public class HelloWorld {
     /* The execution starts here */
    public static void main(String args[])
    {
        System.out.println(“Hello World!“);
     } //end of main()
} //end of class
```

Single line comment
Multi-line comment
entry point for your application
Prints "Hello World!" message to standard output

Type all code, commands and file names exactly as shown. Java is highly *case-sensitive*

Java Development Process



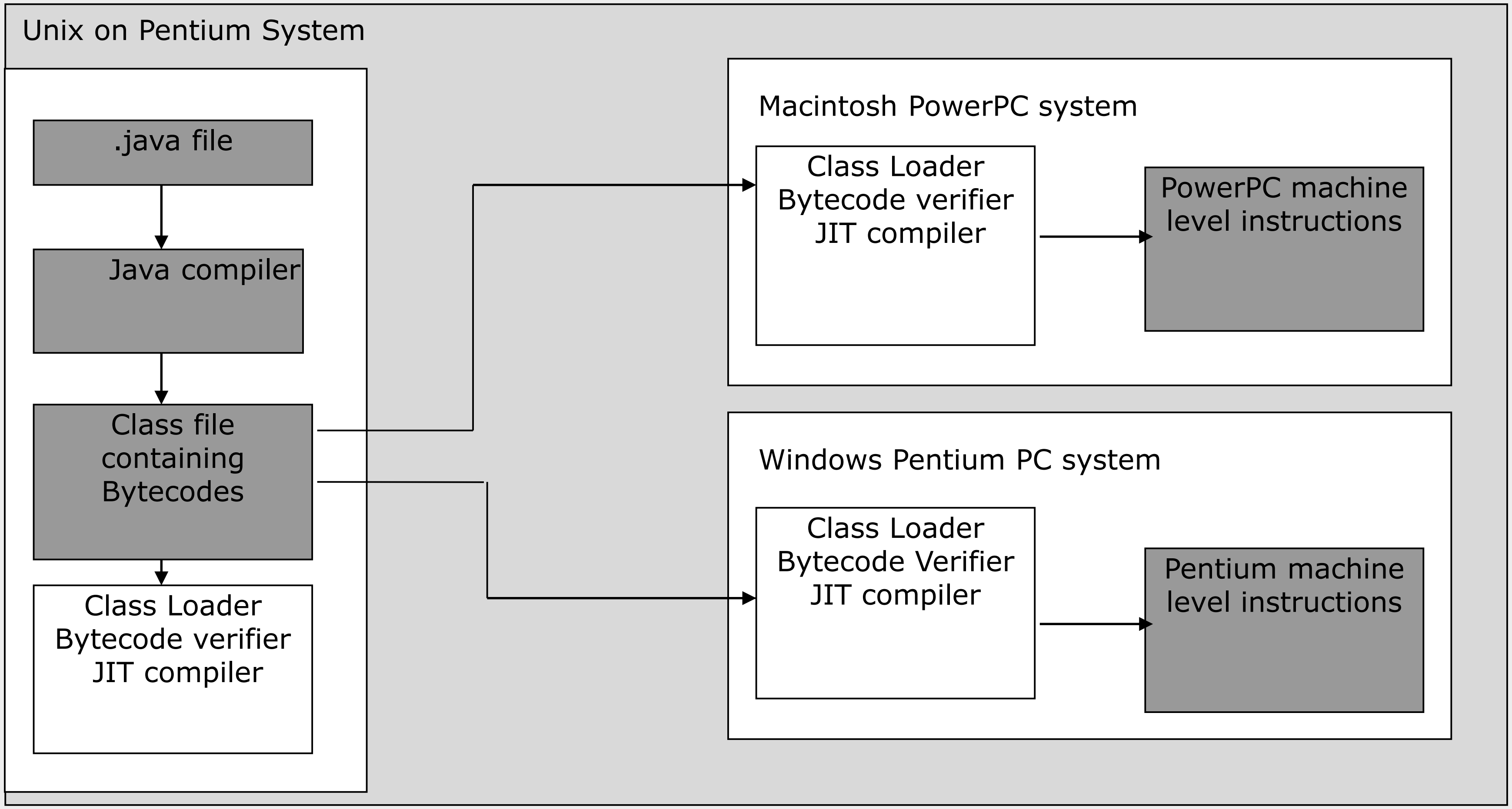


```
F:\JEE-Demos>javac HelloWorld.java

F:\JEE-Demos>java HelloWorld
Hello World!

F:\JEE-Demos>_
```

Demo

- Creating and executing the First Java application

Platform Independence feature of Java

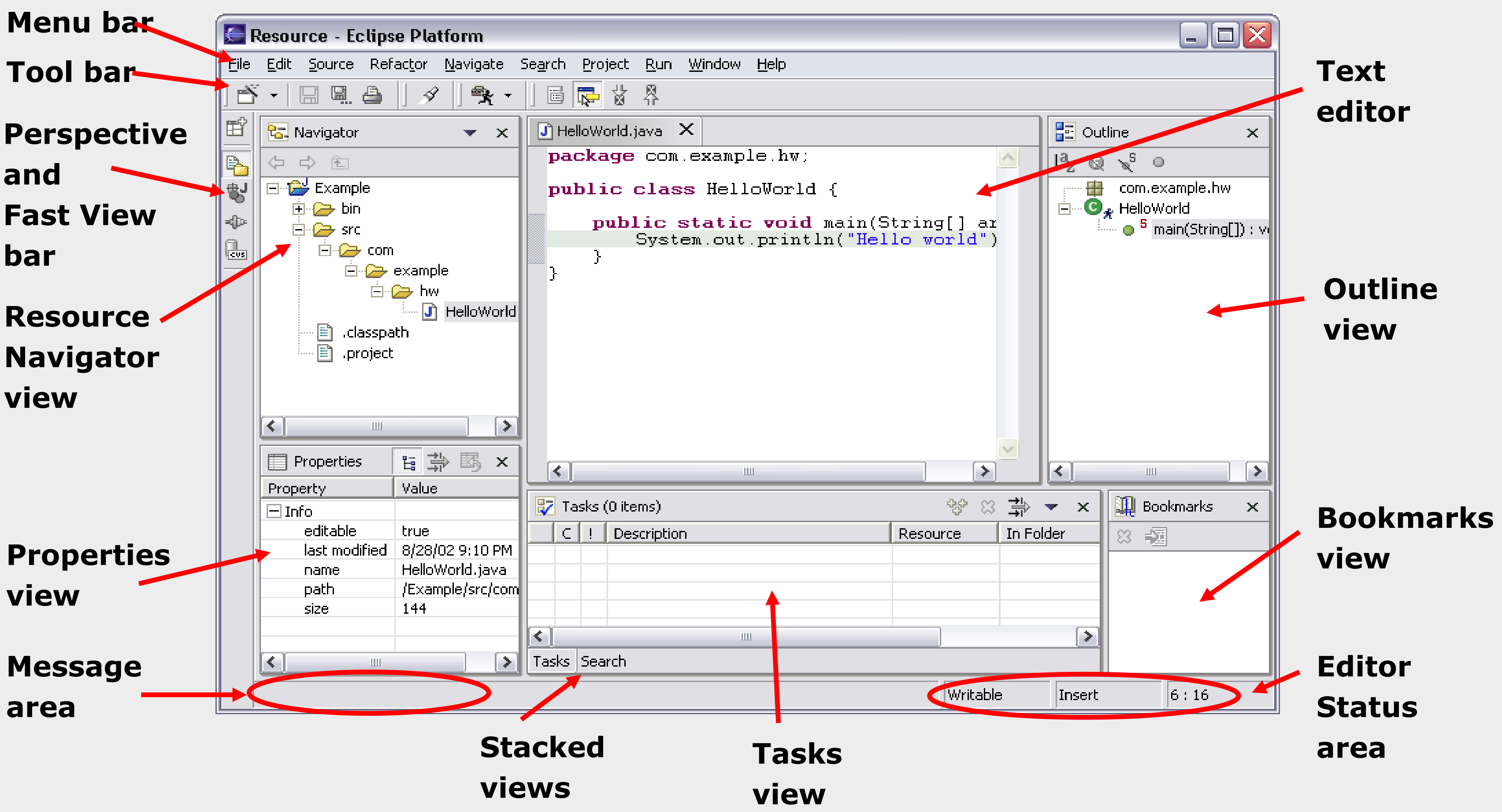


Integrated Development Environment

- IDE is an application or set of tools that allows a programmer to write, compile, edit, and in some cases test and debug within an integrated, interactive environment
- IDE combines:
  - Editor
  - Compiler
  - Runtime environment
  - debugger

2.3.1 : Introduction to Eclipse IDE

Workbench Terminology

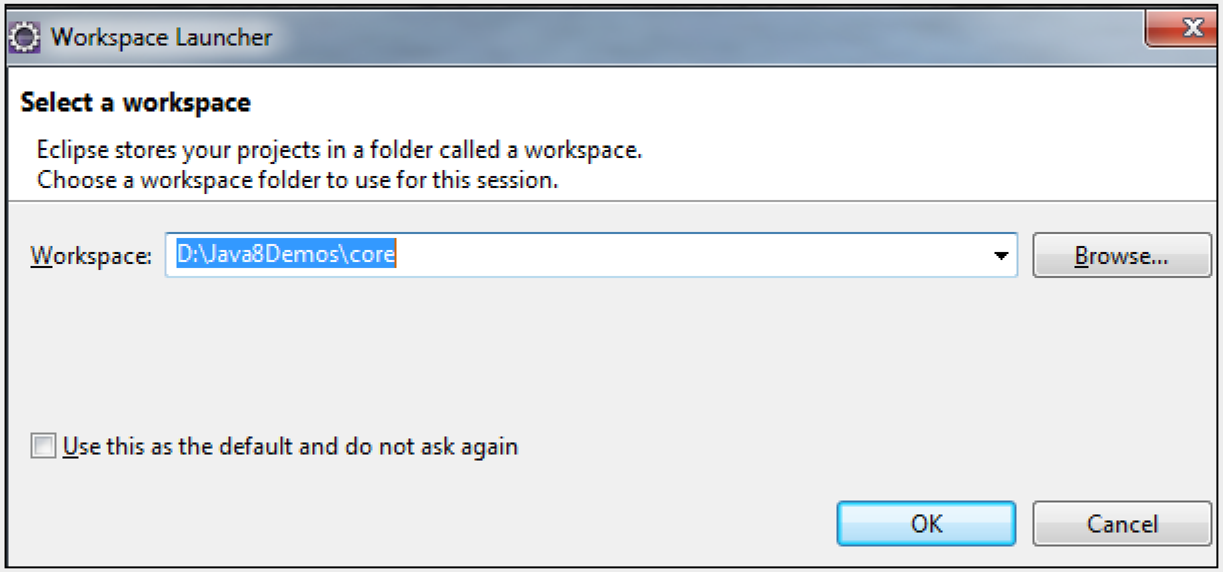


```
package com.example.hw;

public class HelloWorld {

    public static void main(String[] ar
        System.out.println("Hello world")
    }
}
```

Property	Value
Info	
editable	true
last modified	8/28/02 9:10 PM
name	HelloWorld.java
path	/Example/src/com
size	144

Create Workspace

- You need to follow the given steps to create a workspace:
  - Start up Eclipse
  - Supply a path to a new folder which will serve as your workspace
  - The workspace is a folder which Eclipse uses to store your source code


Workspace Launcher

Select a workspace

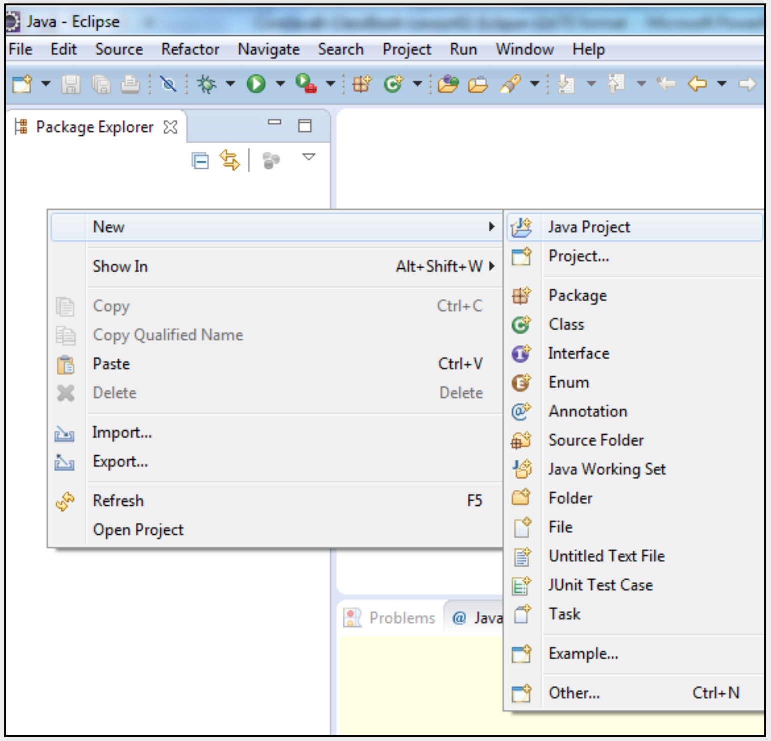
Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

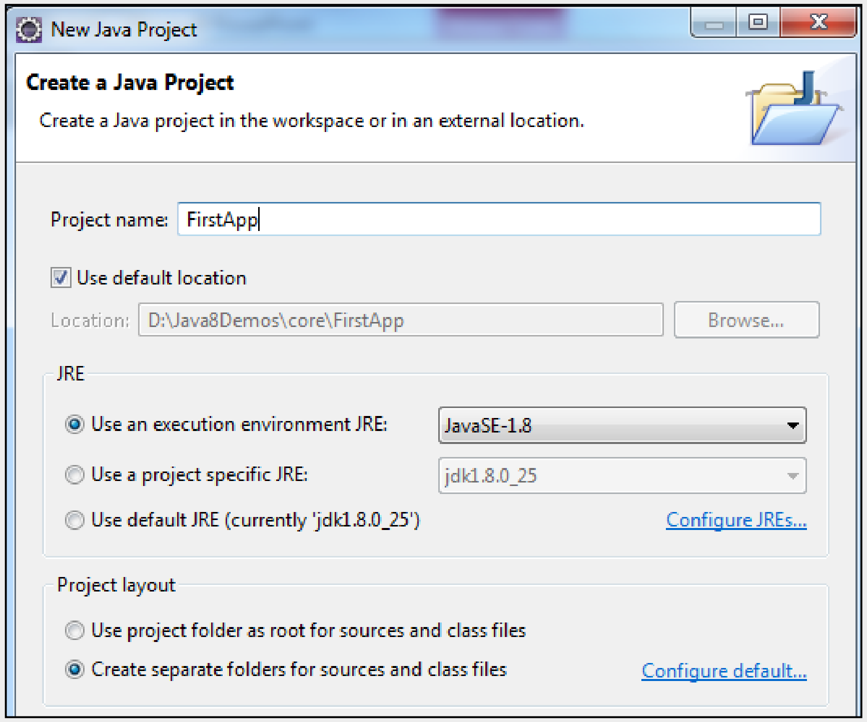
Workspace: D:\Java8Demos\core

☐ Use this as the default and do not ask again

Create a Java Project

- Right-click the Package Explorer panel, and select New-JavaProject.
- Select Java project and provide a Project Name.





New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: FirstApp

[✓] Use default location

Location: D:\Java8Demos\core\FirstApp Browse...

JRE

[✓ ●] Use an execution environment JRE: JavaSE-1.8

○ Use a project specific JRE: jdk1.8.0_25

○ Use default JRE (currently 'jdk1.8.0_25') Configure JREs...

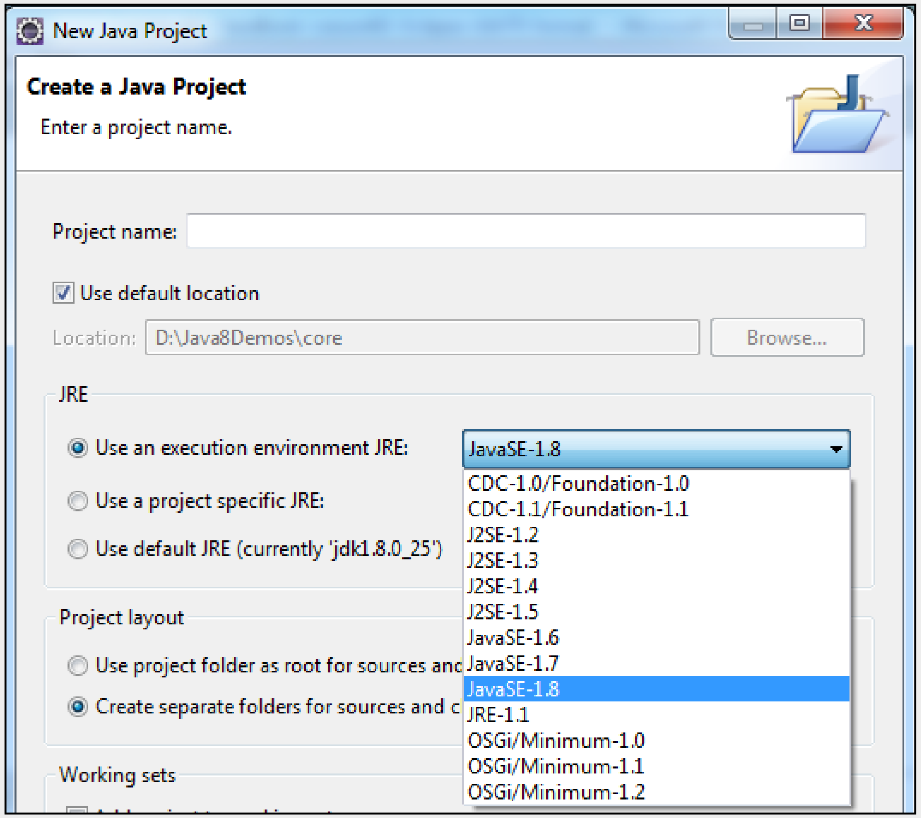
Project layout

○ Use project folder as root for sources and class files

[✓ ●] Create separate folders for sources and class files Configure default...

Select the JRE

- In order to develop code compliant with Java SE 8, you will need a JavaSE-1.8 Java Runtime Environment (JRE)


New Java Project

Create a Java Project

Enter a project name.

Project name:

[✓] Use default location

Location: D:\Java8Demos\core

Browse...

JRE

[✓ Use an execution environment JRE:] JavaSE-1.8

Use a project specific JRE:

Use default JRE (currently 'jdk1.8.0_25')

CDC-1.0/Foundation-1.0
CDC-1.1/Foundation-1.1
J2SE-1.2
J2SE-1.3
J2SE-1.4
J2SE-1.5
JavaSE-1.6
JavaSE-1.7
[✓ JavaSE-1.8]
JRE-1.1
OSGi/Minimum-1.0
OSGi/Minimum-1.1
OSGi/Minimum-1.2

Project layout

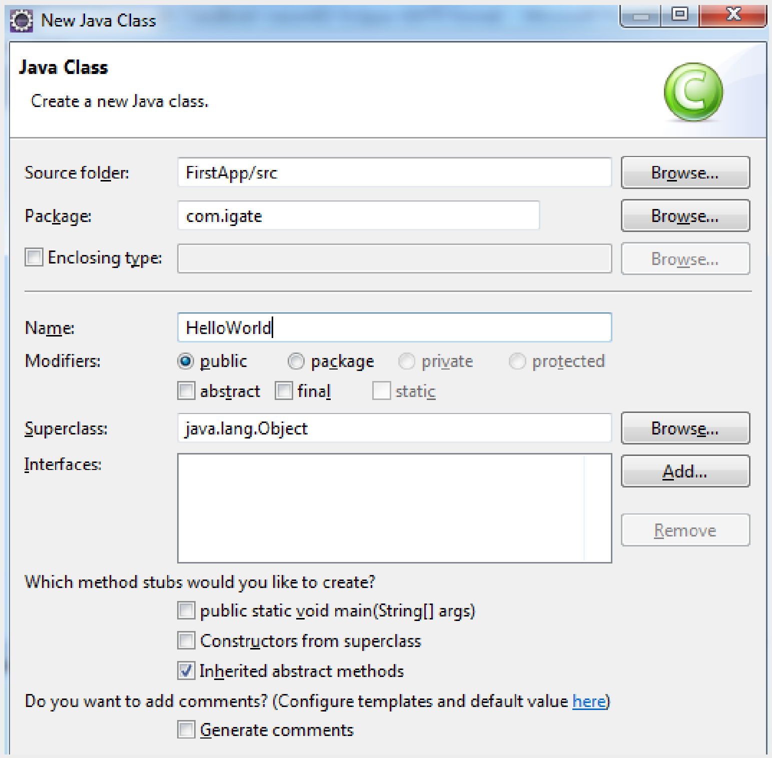
Use project folder as root for sources and

[✓ Create separate folders for sources and c]

Working sets

My first Java Program – Hello World

- Right-click on the project and select "New->Class" Type in your Program code



New Java Class

Java Class

Create a new Java class.

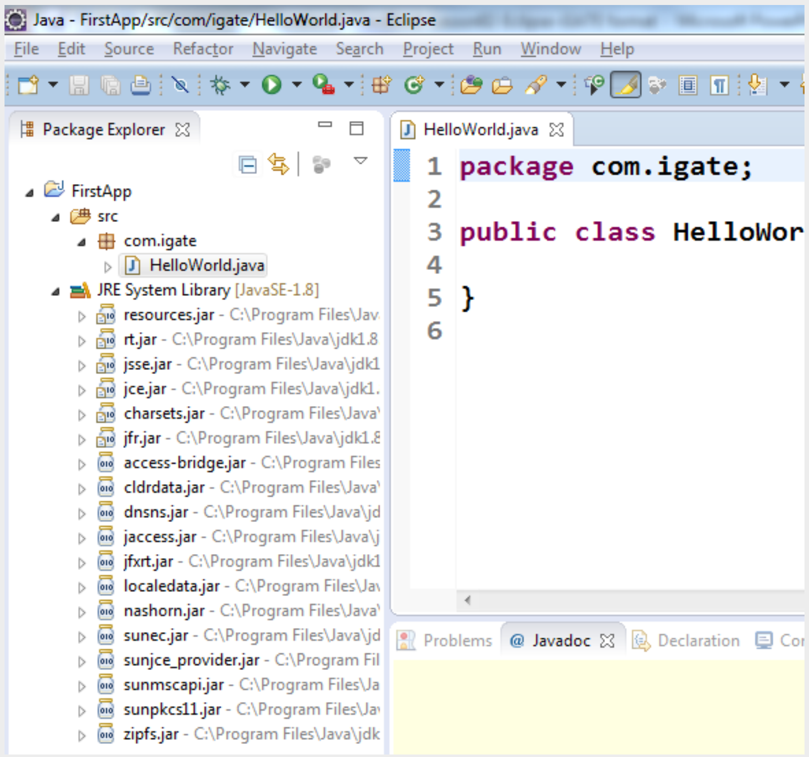
Source folder:	FirstApp/src	Browse...
Package:	com.igate	Browse...
☐ Enclosing type:		Browse...
Name:	HelloWorld	
Modifiers:	[✓ public] package private protected	
	☐ abstract ☐ final ☐ static	
Superclass:	java.lang.Object	Browse...
Interfaces:		Add... Remove

Which method stubs would you like to create?

☐ public static void main(String[] args)
☐ Constructors from superclass
☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

☐ Generate comments



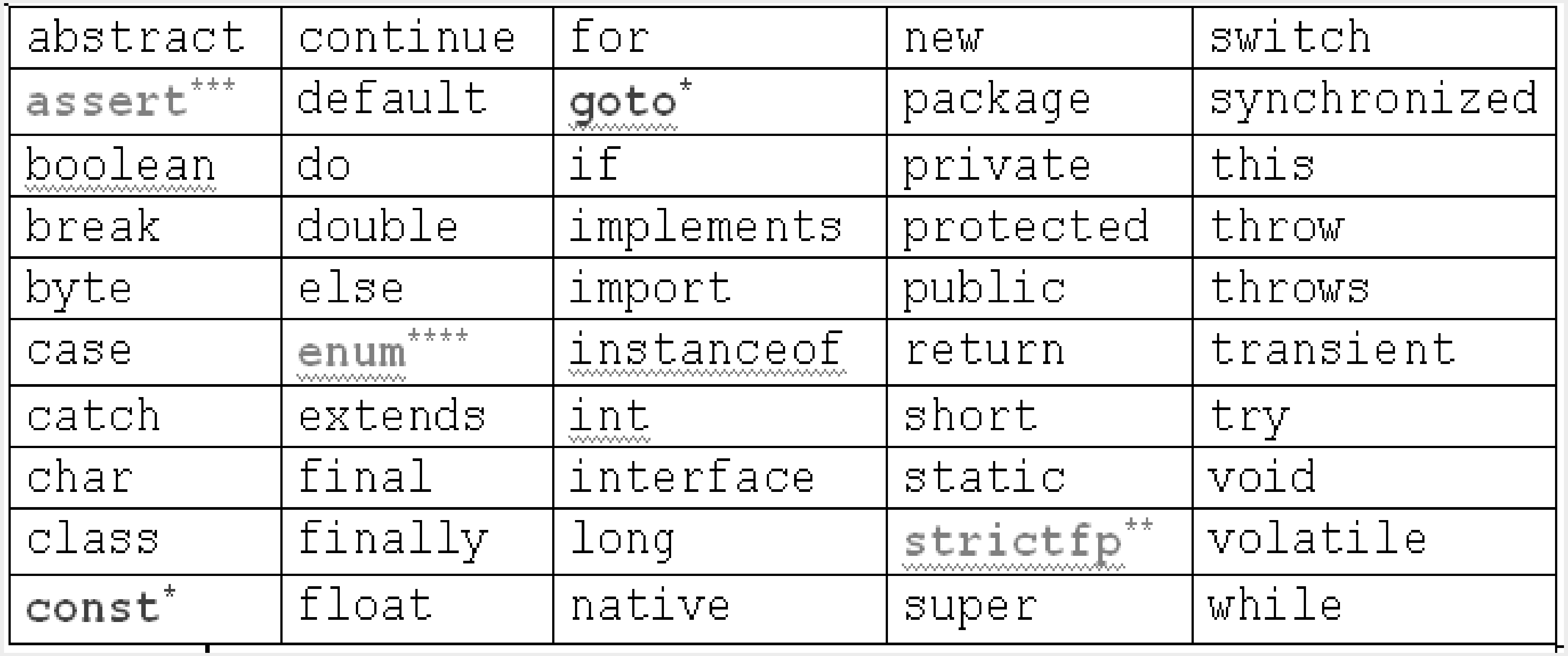
Java - FirstApp/src/com/igate/HelloWorld.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
HelloWorld.java
Problems
@ Javadoc
Declaration
Co

```
FirstApp
  src
    com.igate
      HelloWorld.java
  JRE System Library [JavaSE-1.8]
    resources.jar - C:\Program Files\Jav
    rt.jar - C:\Program Files\Java\jdk1.8
    jsse.jar - C:\Program Files\Java\jdk1
    jce.jar - C:\Program Files\Java\jdk1.
    charsets.jar - C:\Program Files\Java\
    jfr.jar - C:\Program Files\Java\jdk1.8
    access-bridge.jar - C:\Program Files
    cldrdata.jar - C:\Program Files\Java\
    dnsns.jar - C:\Program Files\Java\jd
    jaccess.jar - C:\Program Files\Java\j
    jfxrt.jar - C:\Program Files\Java\jdk1
    localedata.jar - C:\Program Files\Jav
    nashorn.jar - C:\Program Files\Java\
    sunec.jar - C:\Program Files\Java\jd
    sunjce_provider.jar - C:\Program Fil
    sunmscapi.jar - C:\Program Files\Ja
    sunpkcs11.jar - C:\Program Files\Jav
    zipfs.jar - C:\Program Files\Java\jdk
```

```java
package com.igate;

public class HelloWor

}
```

Terms in Java

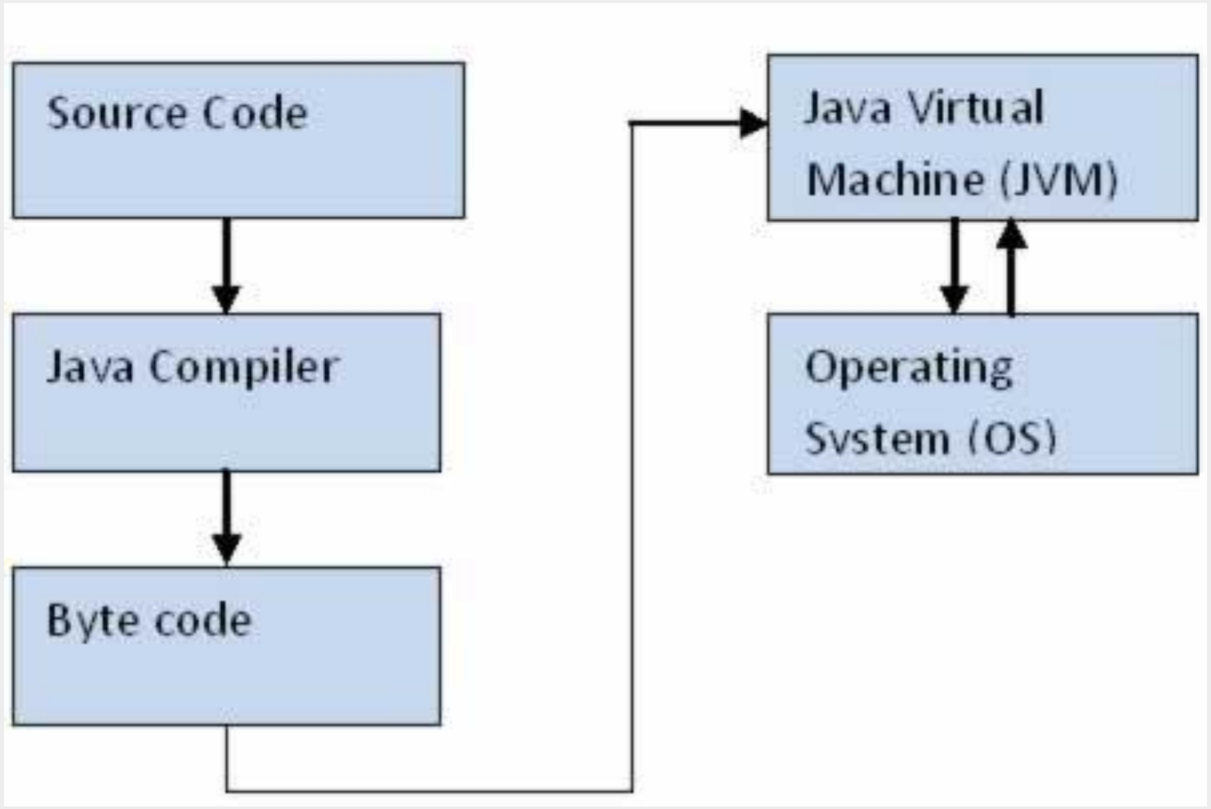


abstract	continue	for	new	switch
assert***	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum****	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp**	volatile
const*	float	native	super	while

JVM Basic Architecture

- It is a specification that provides runtime environment in which java bytecode can be executed.
- JVMs are available for many hardware and software platforms (i.e. JVM is platform dependent).
- The JVM performs following operation:
  - Loads code
  - Verifies code
  - Executes code
  - Provides runtime environment

JVM Basic Architecture



Lab

- Lab 2

Summary

➢ In this lesson, you have learnt:
- How Java is platform Independent
- Writing, Compiling, and Executing a simple program
- Some Important terms in Java
- Integrated Development Environment
- JVM Basic Architecture

Review Question

- Question 1: A program written in the Java programming

  language can run on any platform because...
  - **Option 1:** The JIT Compiler converts the Java program into machine equivalent
  - **Option 2:** The Java Virtual Machine1(JVM) interprets the program for the native operating system
  - **Option 3:** The compiler is identical to a C++ compiler
  - **Option 4:** The APIs do all the work
- Question 2: Java Compiler compiles the source code into ____ code, which is interpreted by ___ to produce Native Executable code.

Review Question

- Question 3: Which of the following are true about JVM?
  - **Option 1:** JVM is an interpreter for byte code
  - **Option 2:** JVM is platform dependent
  - **Option 3:** Java programs are executed by the JVM
  - **Option 4:** All the above is true