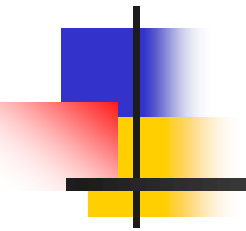
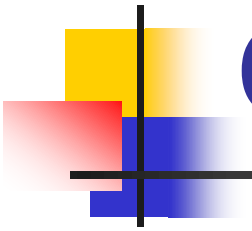


Bài 1. Giới thiệu Java

Lý thuyết và ngôn ngữ hướng đối tượng (bài tập)

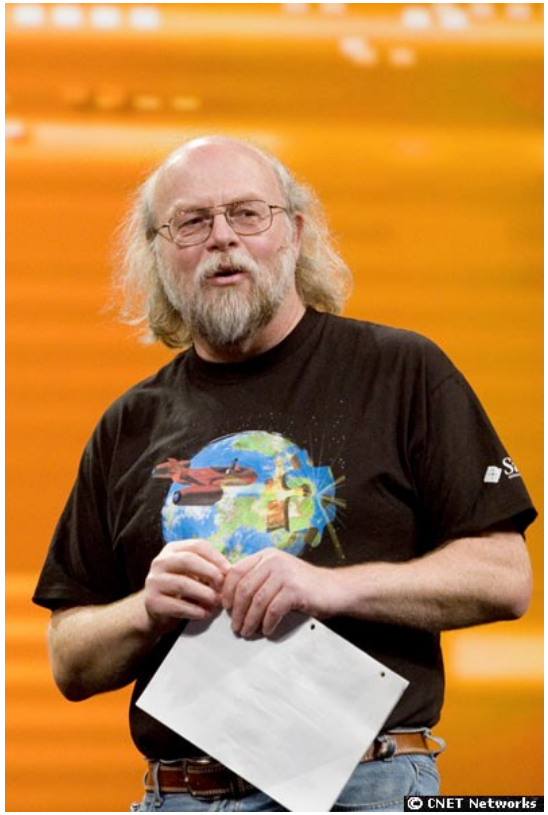


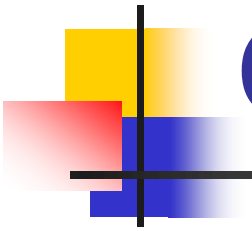
Objectives

- At the end of the lesson, the student should be able to:
  - Describe the features of Java technology
  - Describe the different phases of a Java program

History

- Java
  - was created in 1991
  - by James Gosling et al. of Sun Microsystems.
  - Initially called Oak, in honor of the tree outside Gosling's window, its name was changed to Java because there was already a language called Oak.



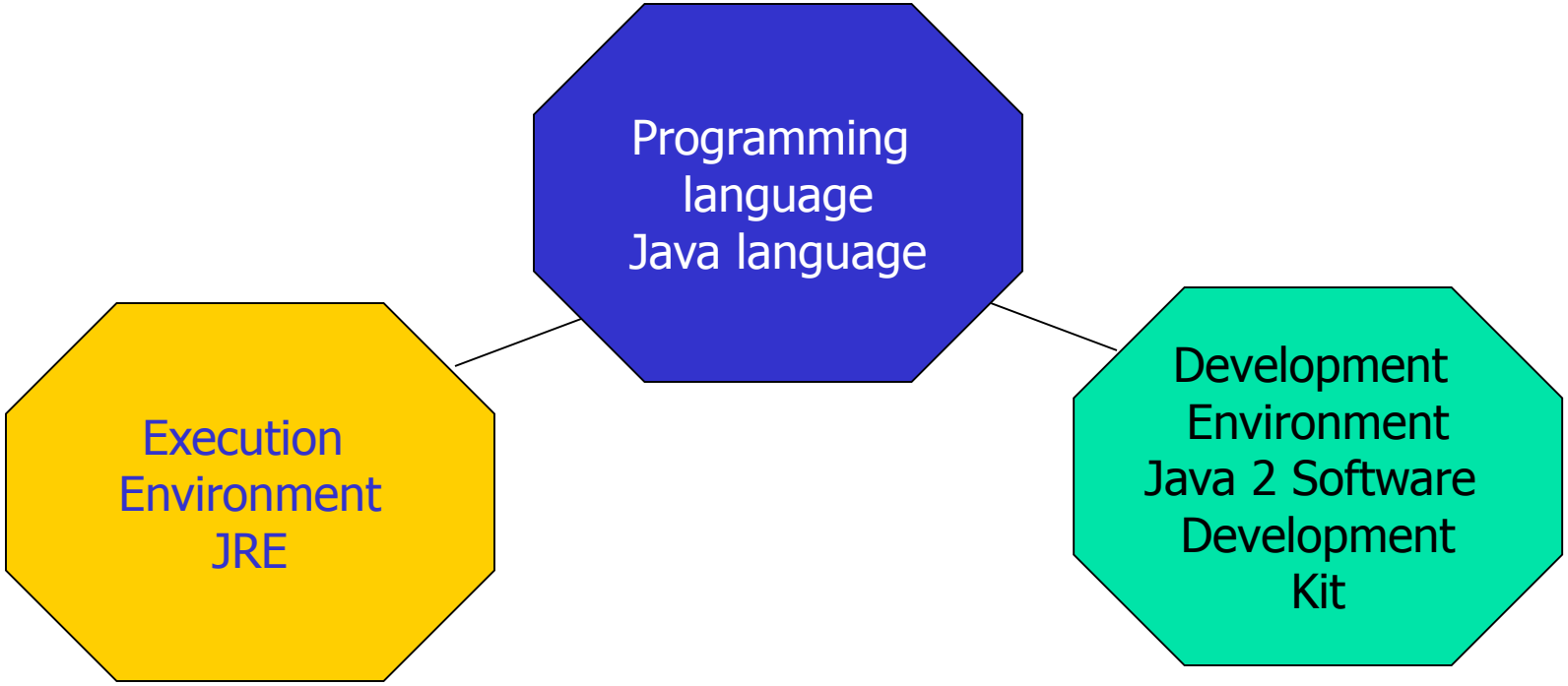


Original motivation

- The need for platform independent language that could be embedded in various consumer electronic products like toasters and refrigerators.
- One of the first projects developed using Java
  - a personal hand-held remote control named Star 7.
- Gosling realized that Java could be used for Internet programming.

Java Technology

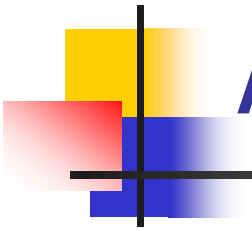
- A programming language
- A development environment
- An execution environment



Programming Language

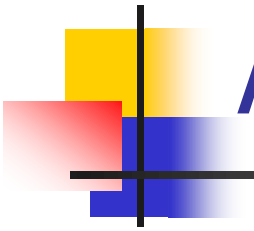
- Java as the language is called the Java language, and is a language that is platform independent and object-oriented.

- As a programming language, Java can create all kinds of applications that you could create using any conventional programming language.



A Development Environment

- As a development environment, Java technology provides you with a large suite of tools (Java 2 SDK):
  - A compiler (javac) and debugger
  - An interpreter (java)
  - A documentation generator (javadoc)
  - A class file packaging tool
  - and so on...

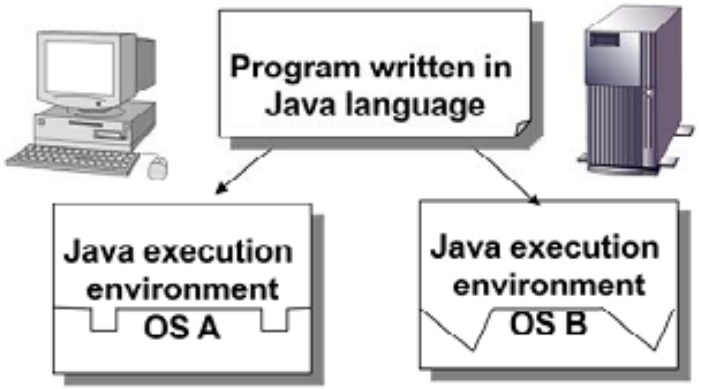


A Execution Environment

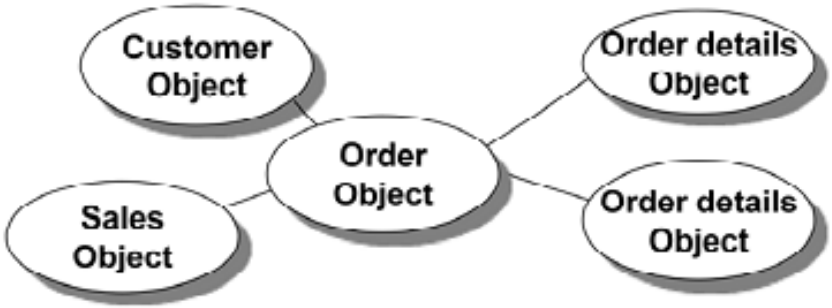
- Java technology applications are typically general-purpose programs that run on any machine where the Java runtime environment (JRE) is installed.
- Two main deployment environments:
  - JRE supplied by Java 2 SDK
  - On your web browser. Most commercial browsers supply a Java technology interpreter and runtime environment.

Java Features

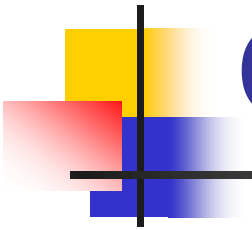
- Platform Independent
- Object Oriented
- Network
- Security
- Simplicity
- Severe error check



Write Once, Run Anywhere

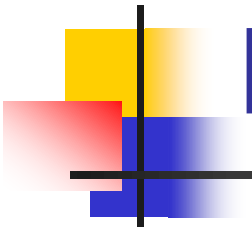


Parts making/reusing by object-oriented



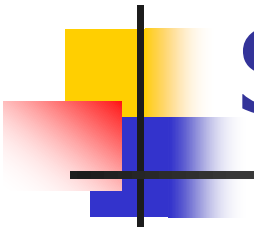
Object – Oriented

- Java is an object-oriented programming language
  - class, object, encapsulation, inheritance, interface,..
- As the term implies, an object is a fundamental entity in a Java program
- Objects can be used effectively to represent real-world entities
- Benefits:
  - Reuse
  - Maintainability and flexibility



Network and Security

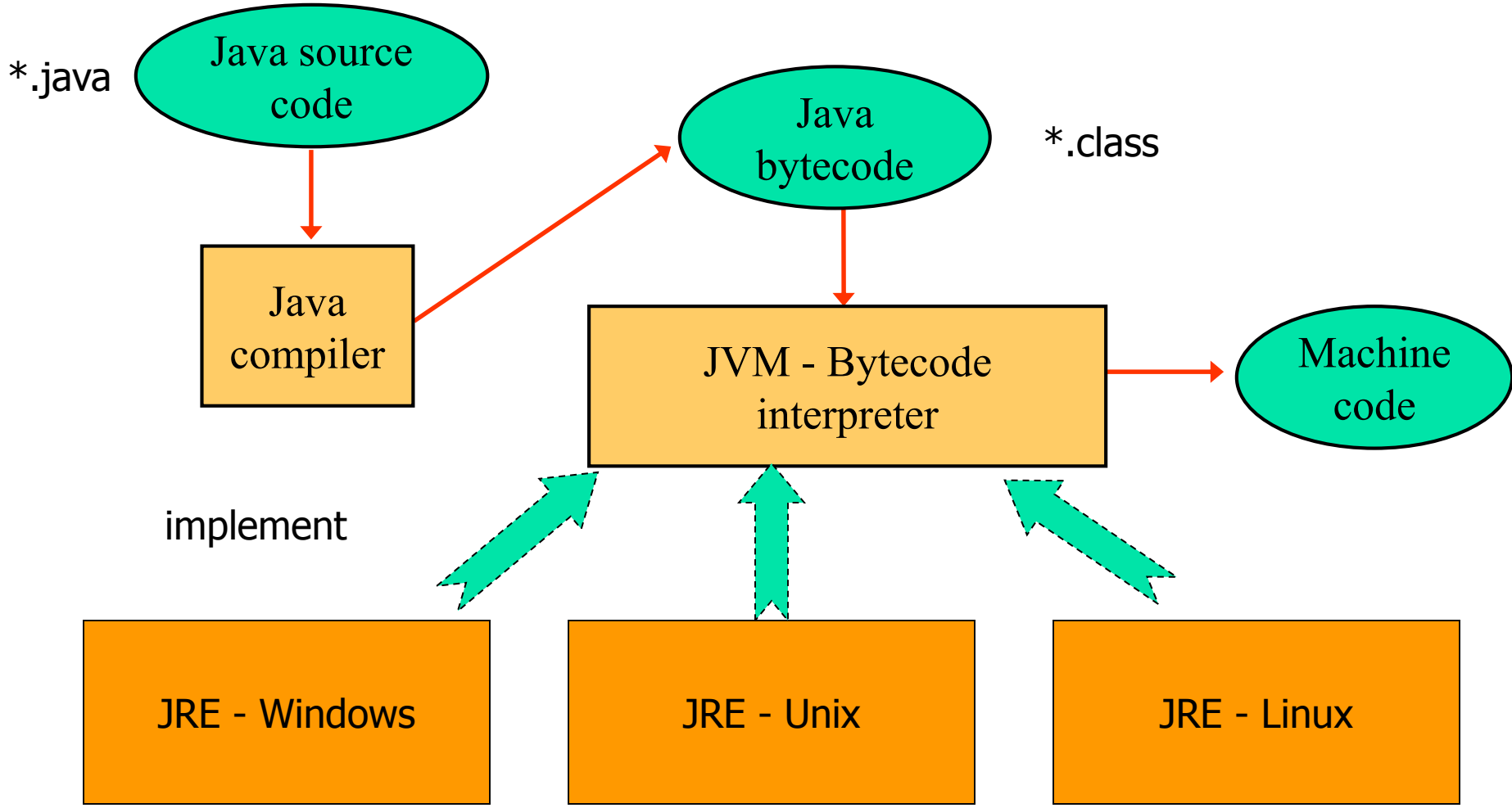
- Network
  - Java support network programming
  - Work with TCP/IP protocol
  - Support Web application (Applet, Servlet)
- Security
  - There are functions that limits the access to the local resource, encryption of information, etc



Simplicity

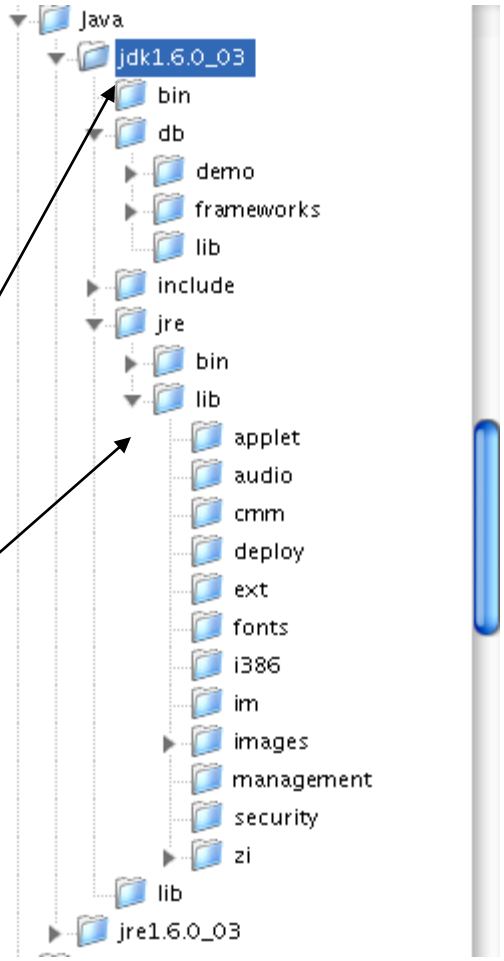
- Easy to write and compile debug
- Automatic support for Memory management
  - Garbage collection: responsible for freeing any memory that can be freed. This happens automatically during the lifetime of the Java program.
  - programmer is freed from the burden of having to deallocate that memory themselves
- Inherit of C/C++ syntax

Java Execution Environment

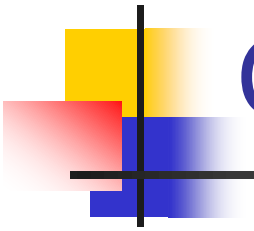


Development Environment of Java

- Compiler `javac.exe`
- Interpreter `java.exe`
- Debugger `jdb.exe`
- Document Generator `javadoc.exe`
- Archiver `jar.exe`
- Class Library `rt.jar`

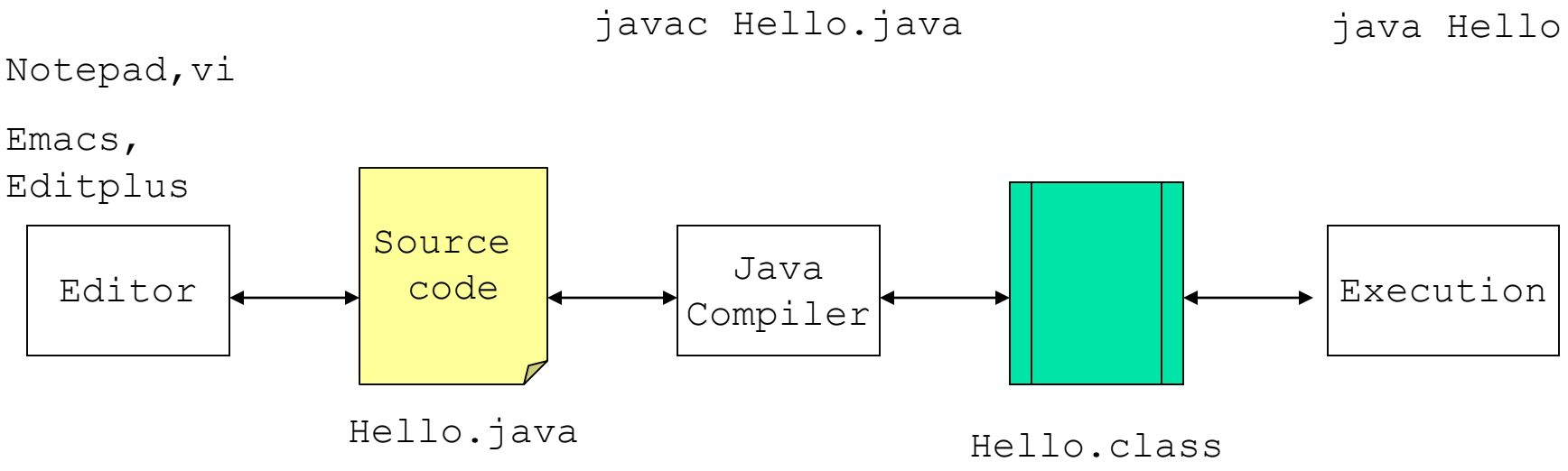


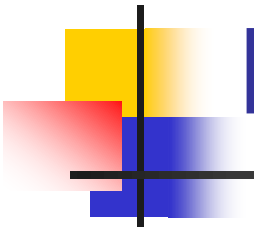
Classification of Java technology:



- J2SE（Java 2 Platform, Standard Edition）
  - Aims at the development of a usual business application. (Have the part that overlaps with J2EE.)
- J2EE（Java 2 Platform, Enterprise Edition）
  - Aims at the development of a decentralized application in a multistory layer in Internet/Intranet.
- J2ME（Java 2 Platform, Micro Edition）
  - Aims at the development of an embedded application such as the cellular phone, the portable terminal, and the microchip, etc.

Making procedure of Java Application





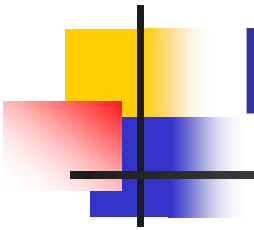
Exercise

- Use your text editor to create this code and save it in the file named Lincoln.java

```
public class Lincoln
{

   public static void main (String[] args)
   {
      System.out.println ("A quote by Abraham
   Lincoln:");

      System.out.println ("Whatever you are, be a good
   one.");
   }
}
```

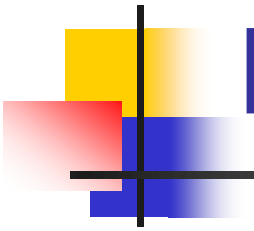


Exercise

- Compile this file by javac command
- Verify if a .class file is produced or not
- Run the class file using java.exe

Brief Introduction about Eclipse

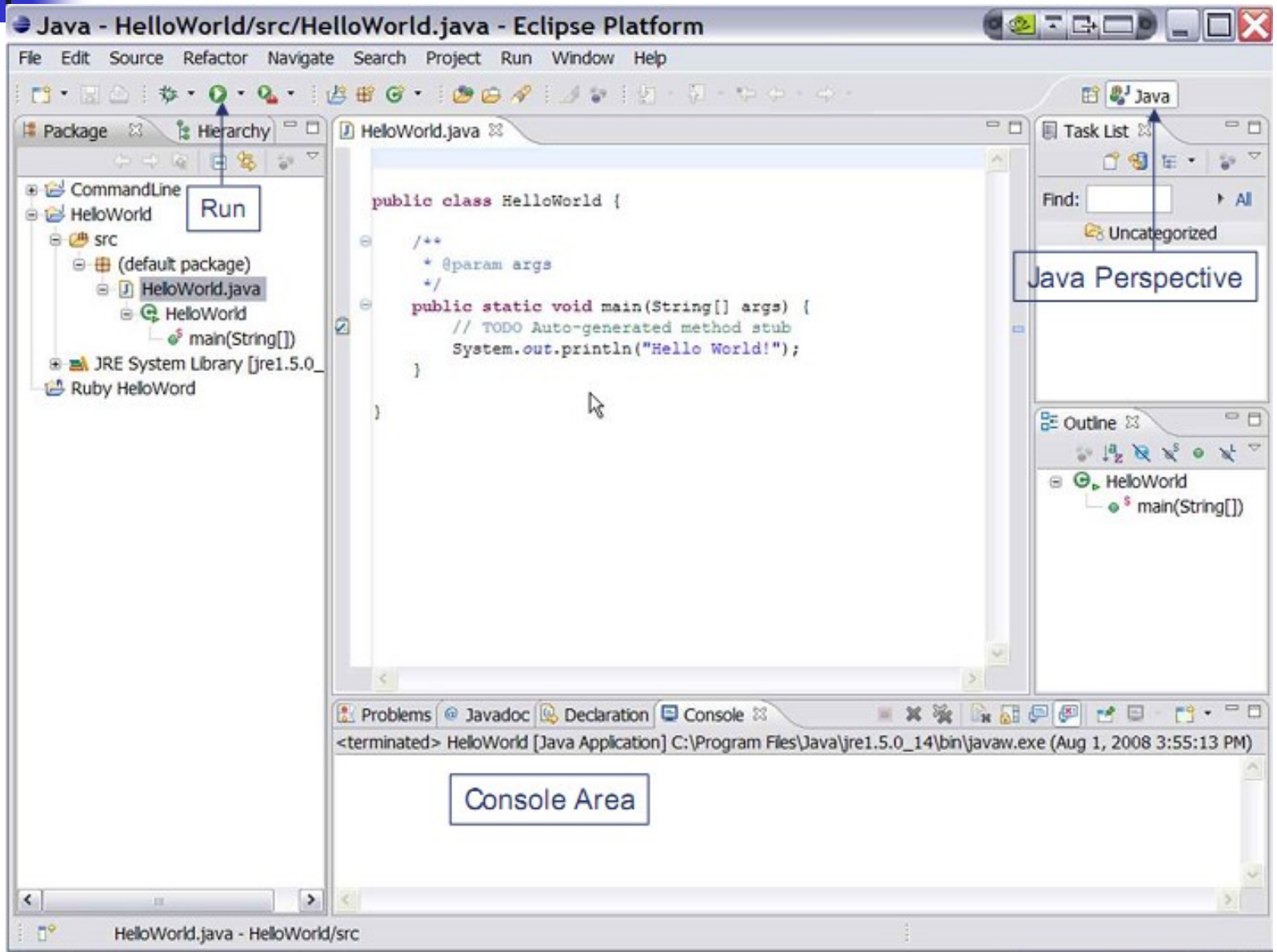
- Eclipse is an open source community whose projects are focused on building an extensible development platform, runtimes and application frameworks for building, deploying and managing software across the entire software lifecycle IDE for java application.
- Java IDE
- Features:
  - Extension of Functions thru plug-in
  - Enhanced development assistance functions
    - (1) Code assistance (Input assistance function)
    - (2) Automatic build function
    - (3) Refactoring
    - (4) Debugger etc.



Eclipse Basics

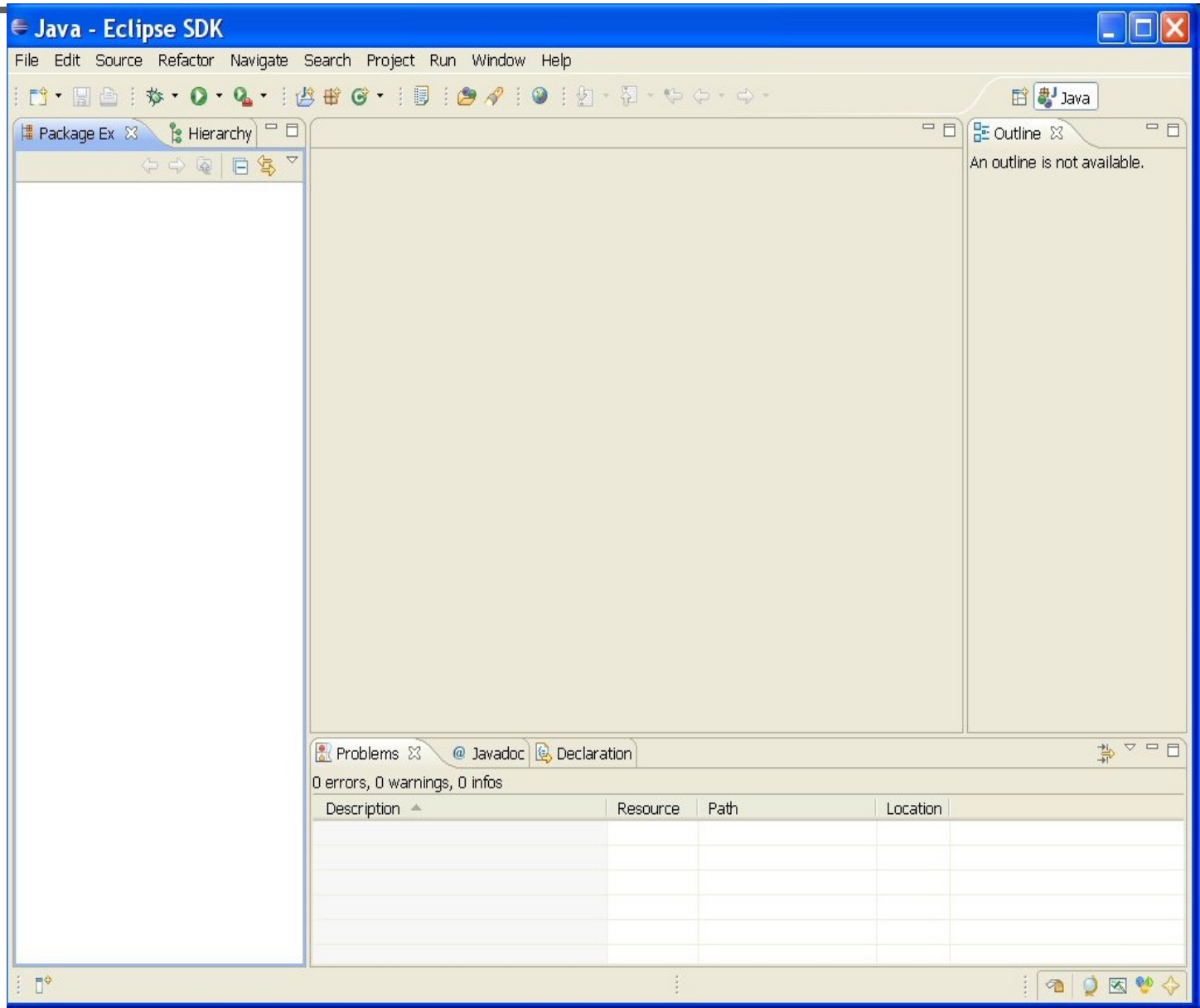
- The Workbench
  - Workbench refers to the desktop development environment
  - Each Workbench window contains one or more Perspectives
  - More than one Workbench window can exist on the desktop at any given time
- Perspectives
  - Contain views and editors
  - Menus and tool bars

Screen composition of Eclipse



Lab: Create – Compile – Run a Java Program with Eclipse

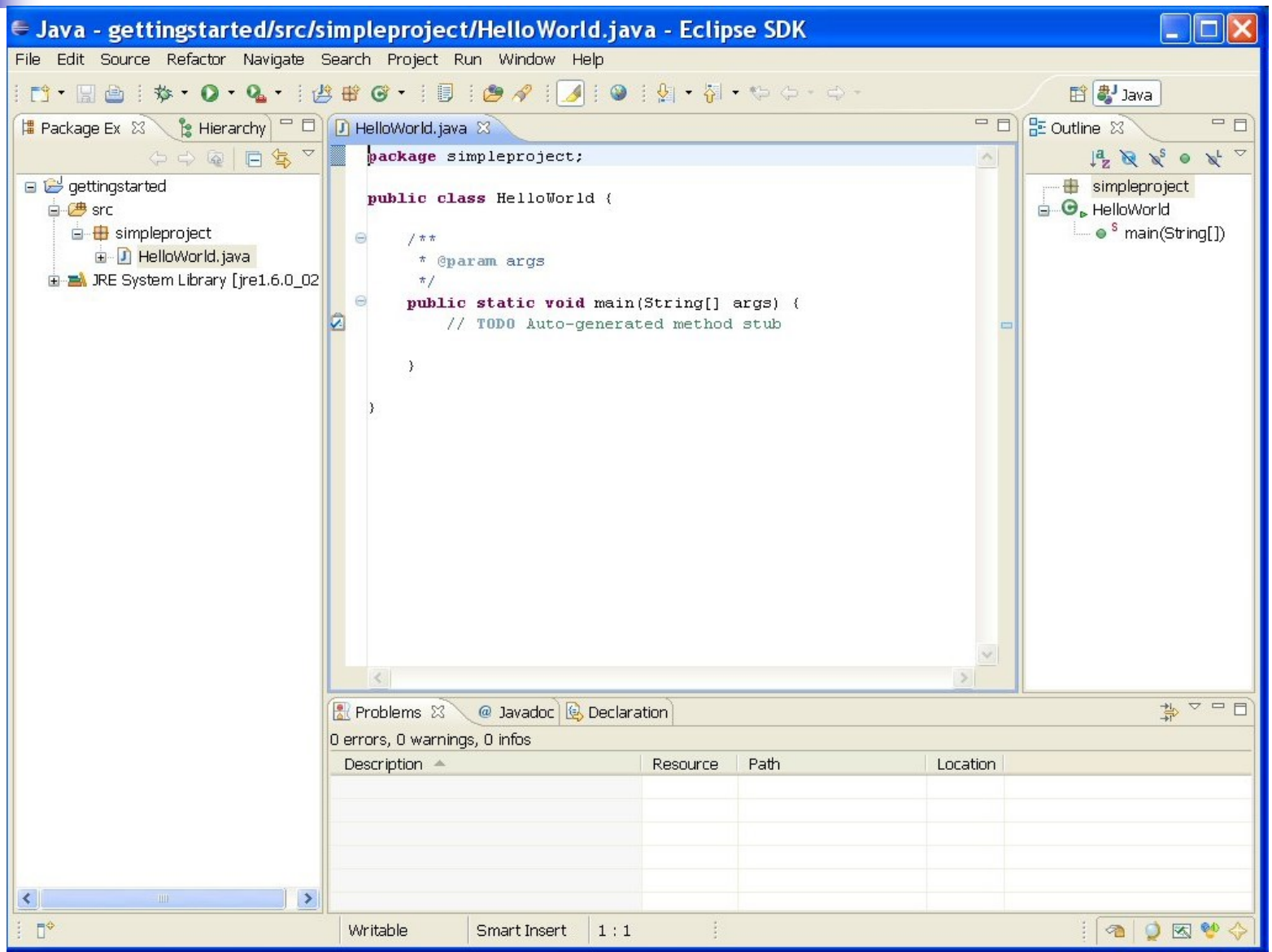
- Starting screen



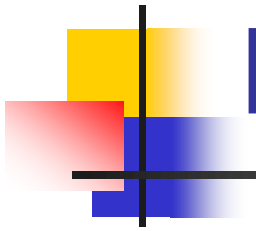
Creating a simple application

- Select **File -> New -> Java Project ->**. Fill in the **Project Name** as `gettingstarted` (all lower case).
- Under Contents, **Create new project in workspace** should be selected.
- Make sure that under **Project Layout** you have chosen **Create separate source and output folders**.
- Click **Finish**
- This has created a **gettingstarted** subdirectory in your workspace. This directory will containing everything you need for this project.
- Create a HelloWorld class by selecting **File -> New -> Class**. This will bring up a **New Java Class** window.
- The source folder should have been filled in with gettingstarted/src. Fill in `simpleproject` as the Package. Use HelloWorld as the **Name**. Pay attention to upper and lower case when you type, as Java is case-sensitive. The **Modifiers** should be **public** and check the box: public static void main ... .
- click **Finish**.

Creating a simple application

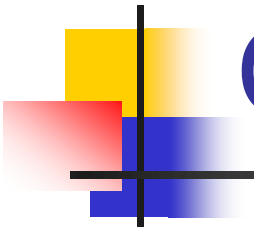


Edit the code



- Delete the comment lines and insert a line so that the main method looks like this:

```
public static void main(String[]
  args) {
     System.out.println("Hello
  world, my name is YOURNAME");
}
```



Compile and Run

- Right click on **HelloWorld** and choose **Run As -> Java Application**
- A **Save and Launch** window may pop up. If it does, select **Always save resources before launching** (so this does not pop up again) and click **OK**. You should see the output in the **Console** window at the bottom of the screen.

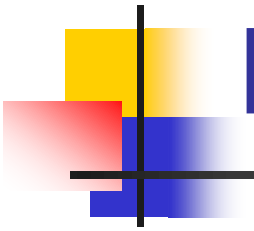
Standard output program

- System.out.println(output data) method
- System.out.print(output data) method
- Use + to unite data

Example

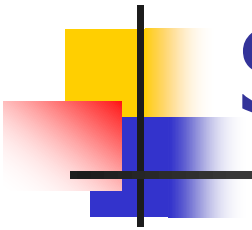
```
System.out.print(“Class: ”);
System.out.println(“2C”);
System.out.print(“Name: ”);
System.out.println(“Xuan Bach”);
System.out.println(“Age:” +20);
```

```
Class: 2C
Name: Xuan Bach
Age : 20
```



Hand on Lab

- Using Eclipse to create a project FirstDay
- Write a program myNeighbour to display the information about the person sitting next to you.



Solution

```
class myNeighbour {
  public static void main(String args[]){

  System.out.println(“My neighbour");
  System.out.println(“Name : Bui Quang”);
  System.out.println(“Age :” + 21);
  System.out.println(“class : 2D”);
  System.out.println(“Promotion :”+ 52) ;

  }
}
```

Ex2 - Dialog

```
// Bai 2:Ghi vao file Dialog.java
import javax.swing.JOptionPane;
public class Dialog{
public static void main(String[] args){
  JOptionPane.showMessageDialog(null,"Xin
  chao ban!");
  System.exit(0);
}
}
```