LESSON I. Introduction

Vu Thi Huong Giang
SoICT (School of Information and Communication Technology)
HUST (Hanoi University of Science and Technology)

Objectives

- Upon completion of this lesson, students will be able to
  - Recall the basics of programming
  - Approach the object-oriented paradigm
  - Understand the Java background
  - Install and use some basic tools for Java programming

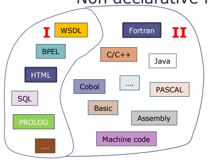
Content

- Programming
- Object-oriented paradigm
- Java background
  - Process of programming using Java technology
  - Java technology
- Basic tools for Java programming

I. Programming

- Given a problem, how to:
  - Design an algorithm for solving it
  - Implement this algorithm as a computer program

→ Needs of programming languages and paradigms
- Language: express the algorithm to a machine
  - Declarative language (**I**): what to do, what to store
  - Non declarative language (**II**): how to do, how to store



I. Programming

- Given a problem, how to:
  - Design an algorithm for solving it
  - Implement this algorithm as a computer program

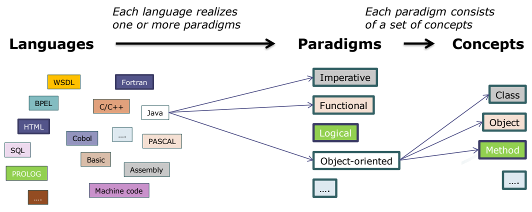
→ Needs of programming languages and paradigms
- Paradigm: comprise a set of concepts that are used as patterns for programming

First do this and next do that	Imperative
Evaluate an expression and use the resulting value for something	Functional
Answer a question via search for a solution	Logical
Send messages between objects to simulate the temporal evolution of a set of real world phenomena	Object-oriented
	....

I. Programming

- Given a problem, how to:
  - Design an algorithm for solving it
  - Implement this algorithm as a computer program

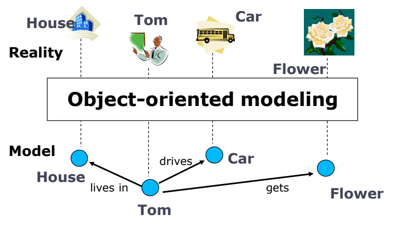
→ Needs of programming languages and paradigms



II. OBJECT-ORIENTED PARADIGM

Object-oriented modeling

- Object in the real world are related to us and each other.
- They can be modeled as software objects



Object

- Object in the real world is represented by:
  - Attributes: information about theirs states
  - Methods: their behaviors related to theirs states.
- Example

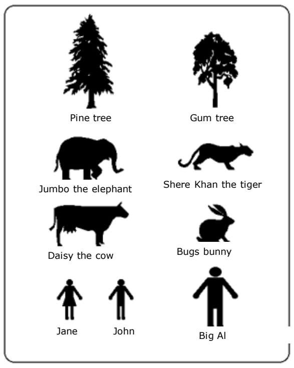
Object	State	Behavior
	- Speedometer: How fast is it moving? - Odometer: How many miles has it driven? - ….	- Move forward - Stop - Reverse - ….
	- Author: Who is the author? - Pages number: How many pages does it contain ? -…	- Buy - Borrow - Count the number of pages….

Class

- A class specifies the common attributes and methods of many individual objects all of the same kind.
- Class is used as the blueprint to create objects

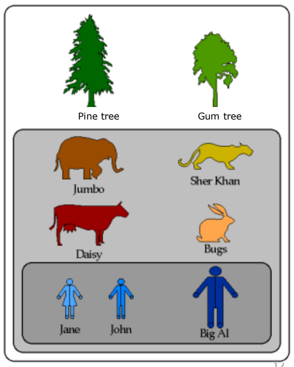
Example

- Objects of the real work that are unclassified



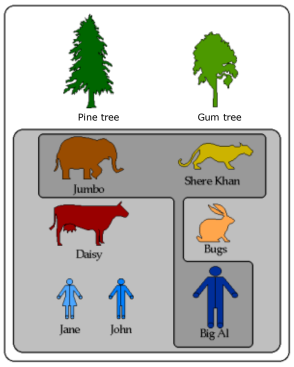
Example: class is used as the blueprint to create objects

- Organisms
- Mammals
- Humans



Example: class is used as the blueprint to create objects

- Organisms
- Mammals
- Dangerous mammals



2. Principles

- Abstraction: Hide details
- Encapsulation: Keep changes local
- Modularity: Control the information flows
- Hierarchy: Order abstractions
- Inheritance: Reuse codes

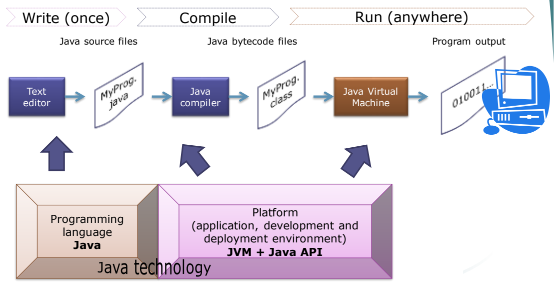
III. JAVA BACKGROUND

1. History
2. Process of programming using Java technology
3. Java technology

1. History

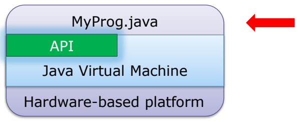
- When and by whom?
  - was created in 1991 by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan of Sun Microsystems.
- Which motivation ?
  - Need of a language, which is independent from platforms and which could be embedded in various electronic devices such as interactive TVs.
- Why Java ?
  - Widely used.
  - Widely available.
  - Embraces full set of modern abstractions.
  - Variety of automatic checks for mistakes in programs.

2. Process of programming using Java technology



a. Java as programming language

- Platform independent and object-oriented programming language
- Able to create all kinds of applications that can be created by any conventional programming language.

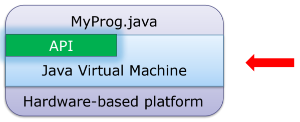


b. Java as platform: JVM + API

- JVM: interpretation for the Java bytecode, ported onto various hardware-based platforms.
- Java API: collection of packages of classes and interfaces providing useful functionalities

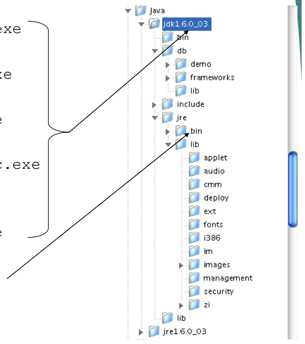
→ These components work as
  - Development environment
  - Application environment and deployment environment

  of Java applications



Development environment

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



Application and deployment environments

- Java programs run on any machine where the Java runtime environment (JRE) is installed.
- 2 main deployment environments:
  - The JRE supplied by the Java Software Development Kit (JDK 7)
  - The Java technology interpreter and runtime environment supplied by commercial web browsers.

Classification of Java platform

- Java SE (Java Platform, Standard Edition)
  - Aims at the development of a usual business application.
- Java EE (Java Platform, Enterprise Edition) and GlassFish
  - Aims at the development of a decentralized application in a multistory layer in Internet/Intranet.
- Java ME (Java Platform, Micro Edition)
  - Aims at the development of an embedded application such as the cellular phone, the portable terminal, and the microchip, etc.
- JavaCard
  - Aims at the development of smart card applications.
- Etc.

IV. BASIC TOOLS FOR JAVA PROGRAMMING

1. Java SE + text editor + console
2. IDE (Eclipse)

1. Java SE + text editor + console

- Java platform standard edition
  - Download the Java SE Development Kit 7 (JDK) at:
  - http://www.oracle.com/technetwork/java/javase/downloads/index.html
  - Don't forget to update the PATH / CLASSPATH environment variables
- Text editor: Notepad, Notepad++, Wordpad, etc.
- Console: for typing Java command line and getting results.

Exercise: first Java program using text editor + console

- Use your text editor (e.g Notepad) to create this code and save it in the file named SayHello.java

```
public class SayHello {
  // The program starts here
  public static void main (String[] args) {
     // print "Chao!" on the screen
     System.out.println ("Chao!");
  }
}
```

Exercise: first Java program using text editor + console

- Compile this file by **javac** command

```
> dir
SayHello.java
> javac SayHello.java
```

- Verify if a **.class** file is produced or not

```
> dir
SayHello.java
SayHello.class
```

- Run the class file using **java** command

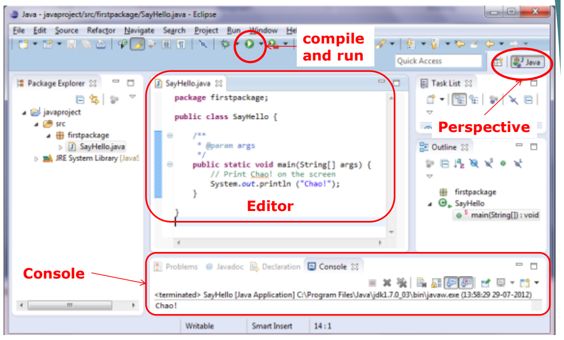
```
> java SayHello
Chao!
```

2. Eclipse (Juno 4.2)

available at: http://www.eclipse.org/downloads/

- Open source Java IDE
- Features:
  - Extension of functions through plug-ins
  - Enhanced development assistance functions: Code assistance, automatic build function, refactoring, debugger, etc.
- Basics:
  - Workbench: desktop development environment, each contains one or more Perspectives
  - Perspectives: Contain views and editors, menus and tool bars

Screen composition of Eclipse

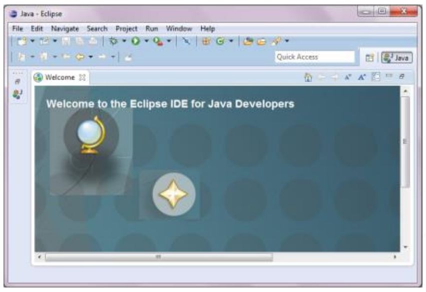


```
package firstpackage;

public class SayHello {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // Print Chao! on the screen
        System.out.println ("Chao!");
    }

}
```

```
<terminated> SayHello [Java Application] C:\Program Files\Java\jdk1.7.0_03\bin\javaw.exe (13:58:29 29-07-2012)
Chao!
```

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

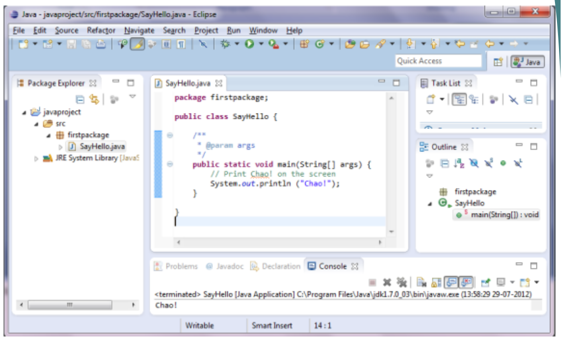
- Starting screen


Java - Eclipse
File Edit Navigate Search Project Run Window Help
Quick Access
Java
Welcome
Welcome to the Eclipse IDE for Java Developers

Creating a simple application

- Select File -> New -> Java Project ->. Fill in the Project Name as javaproject
  - Under Contents, select Create new project in workspace
  - Under Project Layout , choose Create separate source and output folders
  - Click Finish
- Create a SayHello class by selecting File -> New -> Class. This will bring up a New Java Class window.
  - Fill in javaproject /src as Source folder.
  - Fill in firstpackage as Package.
  - Use SayHello as the Name.
  - Select public static void main … .
  - Click Finish.

Creating a simple application



```
package firstpackage;

public class SayHello {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // Print Chao! on the screen
        System.out.println ("Chao!");
    }

}
```

```
<terminated> SayHello [Java Application] C:\Program Files\Java\jdk1.7.0_03\bin\javaw.exe (13:58:29 29-07-2012)
Chao!
```

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) {
  // Print Chao! on the screen
  System.out.println ("Chao!");
}
```

Compile and Run

- Right click on SayHello 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.

Quiz

1. Java program is termed “Write once, run everywhere”. Explain.
2. Give an example of class and objects in the real world.
3. Write a program named MyFavouriteBook to display the information about the book you love (title, author, language) and why you love it.
4. Using javac and java command to compile and run it
5. Using Eclipse to create a project FirstLecture, them compile and run it.

Solution

Quiz 1

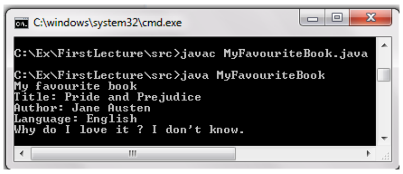
- A Java program can be written on any device, for example a PC.
- Then, it is compiled into a standard byte code and be expected to run on any device such as cell phone, mainframe without any adjustments, if these devices are equipped with a Java virtual machine (JVM)

Quiz 2

- Consider your marker pen. Each marker pen contains the same components, so we can say that each marker pen was manufactured from the same blueprint.
- Your marker pen (a specific pen object) is an instance of a class of objects known as marker pens. You can easily describe the state and behavior of a marker pen.

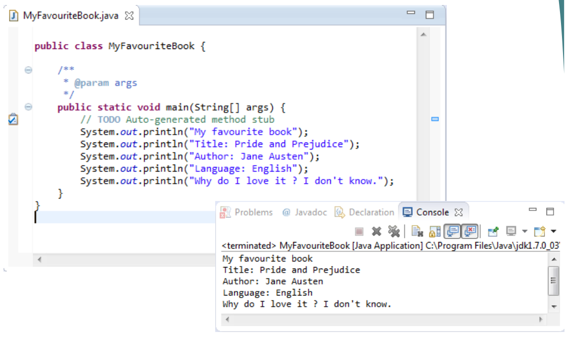
Quiz 3-4: Solution

```
public class MyFavouriteBook {
    public static void main(String[] args) {
      System.out.println("My favourite book");
      System.out.println("Title: Pride and Prejudice");
      System.out.println("Author: Jane Austen");
      System.out.println("Language: English");
      System.out.println("Why do I love it ? I don't know.");
    }
}
```



```
C:\Ex\FirstLecture\src>javac MyFavouriteBook.java

C:\Ex\FirstLecture\src>java MyFavouriteBook
My favourite book
Title: Pride and Prejudice
Author: Jane Austen
Language: English
Why do I love it ? I don't know.
```

Quiz 5: Solution



```java
public class MyFavouriteBook {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("My favourite book");
        System.out.println("Title: Pride and Prejudice");
        System.out.println("Author: Jane Austen");
        System.out.println("Language: English");
        System.out.println("Why do I love it ? I don't know.");
    }
}
```

```
<terminated> MyFavouriteBook [Java Application] C:\Program Files\Java\jdk1.7.0_03
My favourite book
Title: Pride and Prejudice
Author: Jane Austen
Language: English
Why do I love it ? I don't know.
```

Review

- Programming
  - Language
  - Paradigm
- Object-oriented paradigm
  - Object: state + behavior
  - Class: blueprint for creating objects
  - Principles: abstraction, encapsulation, hierarchy, modularity, inheritance
- Java background:
  - Language: Java
  - Platform: JVM + API
- Basic tools for Java programming
  - Platform +Text editor + console
  - Platform + IDE