# Object-oriented Software Design and Development CCP114N

Week 2

Overview of the Java programming language
Review of Java basics
Simple Input and Output

## Java – an intro

- Java has been developed at Sun Microsystems
- Now it becomes a ‘universal’ OO programming language to develop a wide range of software from embedded to desktop to client-server and to internet applications.
- Among others are
  - simple
  - object-oriented
  - Java is both compiled and interpreted

## Java is an object-oriented programming language

- Java programs are composed of Classes
  - no ‘stand-alone’ procedures/functions
  - a Java application is a class
  - classes are ‘object factories’
- In Java, a Class:
  - is the basic unit of compilation
  - contains
    - methods or class *member functions*
    - data items or class *data members*

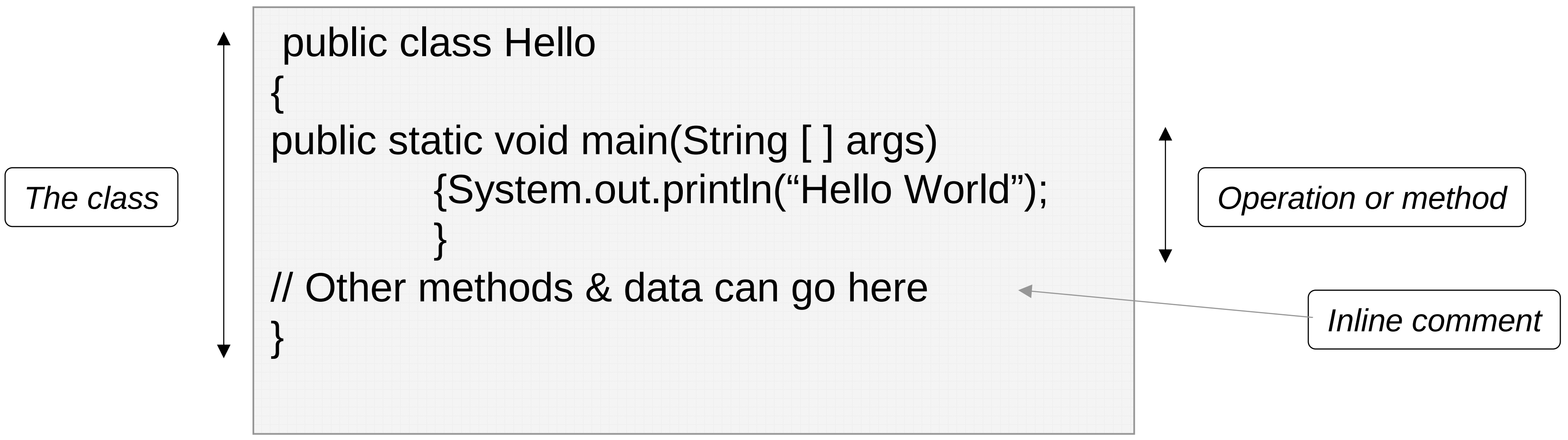
## Basic Structure of a Java Program

- An application
  - ❑ is a class with a main( ) method
  - ❑ is started at main( )
  - ❑ has only one main( ) method

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

## Basic Program Structure

- A class contains
  - ❑ an internal data structure
  - ❑ operations to manipulate this data
    - operators
    - methods



```
 public class Hello
{
public static void main(String [ ] args)
              {System.out.println(“Hello World”);
              }
// Other methods & data can go here
}
```

## Java basics: code

- Statements
  - ❑ may be *blocks* i.e. {a sequence of statements}
  - ❑ ended with a semicolon (;)
- Expressions
  - ❑ Data e.g. constants, variables, other expressions
  - ❑ Operators e.g. +, -, ++
- Comments: useful for someone else to understand your code

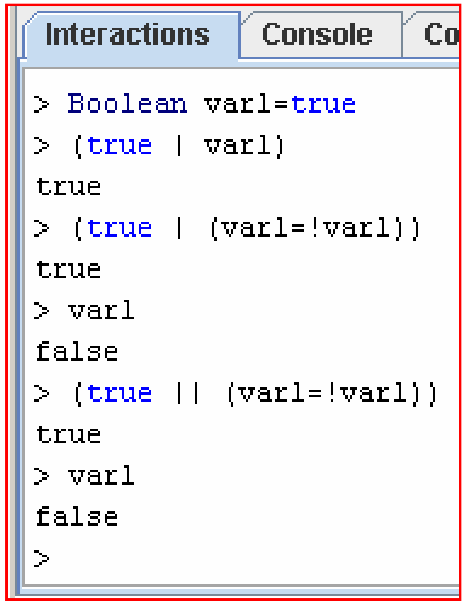
```
// A single line comment
/* simple comments example*/
/** two star comment is picked by javadoc tool */
/** A multiple line
    comments example
*/
```

## Java basics: data

- Primitive (built-in) types
  - 4 integer types: **byte** (8 bits), **short** (16 bits), **int** (32 bits), **long** (64 bits)
  - 2 floating point types: **float** 32 bits; **double** 64 bits
  - Characters
    - **char**
    - in Java Unicode is used ( NB: not ASCII)
    - a character variable contains a short (a 16-bit) integer value
  - **Boolean**: true, false
- Your can define your own data types based on basic built-in types as programmer-defined classes
- Declaring variables `type identifier`
  - `int monthNumber;`
  - `char YesNo;`

## Java basics: operators

- Assignment, e.g.
  - `int i=0; i=i+1;`
- Operators
  - binary operator: *, /,+,-,% (modulus)
  - unary operators: - (negation), ++(increment), -- (decrement)
  - logical operators e.g.: ! (not), & (and), | (or)
    - ❑ see the difference between | and | |
  - others: (for composing logical expressions)
    - ❑ <, >, <=, >=
    - ❑ == (equal)
    - ❑ != (not equal)



```
> Boolean var1=true
> (true | var1)
true
> (true | (var1=!var1))
true
> var1
false
> (true || (var1=!var1))
true
> var1
false
>
```

## Program Flow Control

- Control Statements
  - Selection
    - ❑ Simple If (expr1) statement; or more complex e.g.

```
If (<boolean expression>) {<statements1>} else {<statements2>}
```

```
switch (<expr>) { case cexp1: <statements1> case: cexp2<statements2>...}
```

  - Iteration

```
for (<expr1>;<boolean expression>;<expr2>) {<statements>}
```

```
while (<boolean expression>) {<statements>}
```

```
do{<statements>} while (<boolean expression>);
```

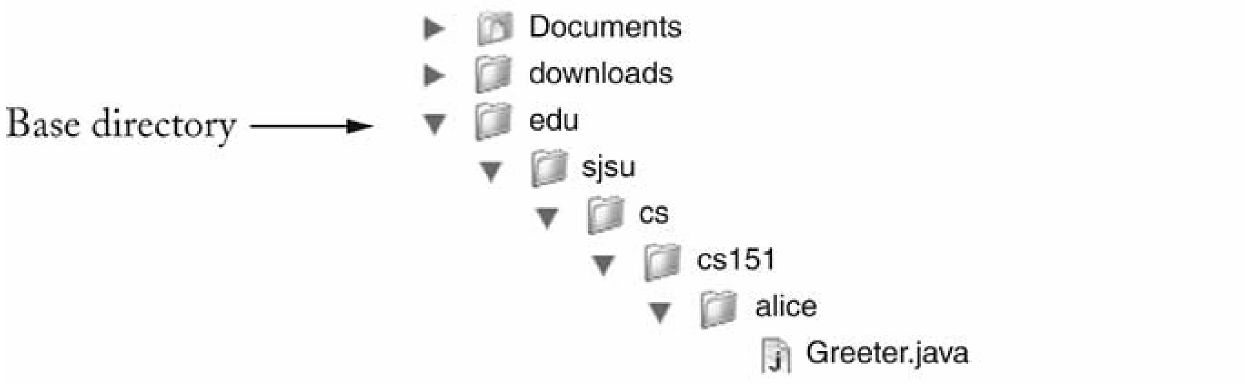
## Class and Objects

- Class and Object are fundamental constructs in Java
  - A program (application or applet) is a class
    - A running instance of the program is an object of the program’s class
  - Data are objects of pre-defined classes
    - int RoomNumber = 15;
    - String BuildingName = “GraduateCentre”;
  - Variable holds a reference to an object, but not the object itself
    - This can be seen using a debugger e.g in BlueJ
    - Same object may be referred to with multiple variables
- The null reference: it refers to no object
  - BuildingName = null;
- The this reference: refers to implicit object of the current class

```
public class Room ( ) {
    private int roomNo;
    public Room(int anInteger)
      { this.roomNo = anInteger;   }
}
```

## Packages

- Packages are used to manage Java applications with many classes just like files are organised into folder/directory tree
- Classes are grouped into packages



- Package names are dot-separated identifier sequences
  java.util
  javax.swing
  com.sun.misc
  edu.sjsu.cs.cs151.alice

## Packages (cont’)

- If your class uses Java classes from the SDK they should be imported at the top of your class’ code.
  - import java.util.ArrayList;
    . . .
    ArrayList a; // i.e. java.util.ArrayList
  - The java.lang package is the default package of the language, hence no need to import it.
- You can also create package for multiple class application (as required in JBuilder and NetBean)
  - The package name should be at the top of every class of your app.
    - E.g. package TimeTabling;
    - Package name must match subdirectory name
  - Full name of class = package name + class name
  - Class without package name is in "default package"

## Simple data input and output

- Use the component Swing JOptionPane

  ```
  import javax.swing.JOptionPane;
  ```

- Using input dialog for accepting user input

  ```
  String input = JOptionPane.showInputDialog("How old are
  you?");
  ```

  If user cancels, result is null else convert the inputted string to an integer

  ```
  if (input != null) age = Integer.parseInt(input);
  ```

- Using message dialog for output

  ```
  String outputString = “The number inputted is “ + age;
  JOptionPane.showMessageDialog(null, outputString);
  ```

## Summary

- Features of the Java language
- Java basic programming constructs
- Simple input and output