Introduction to Java

Zhifu Pei
CSCI5448 – Spring 2011
Prof. Kenneth M. Anderson
Overview

- **Introduction**
  - History, Characteristics of Java language

- **Java Language Basics**
  - Data types, Variables, Operators and Expressions

- **Anatomy of a Java Program**
  - Comments, Packages, Classes, Reserved Words, Modifiers, Blocks, Statements, Methods and main Method

- **Concepts of Object-Oriented Programming**
  - Objects, Classes, Inheritance, Interface and Package
History

- Developed by James Gosling at Sun Microsystems.
- Released by Sun Microsystems in 1995.
- Hotjava – The first java-enabled web browser
- J2EE, J2ME and J2SE
Characteristics

- Simple, object-oriented, and familiar
- Robust and secure
- Architecture-neutral and portable
- High performance
- Interpreted, threaded, and dynamic
A data type is a scheme for representing values.
- Values are not just numbers, but any kind of data that a computer can process.

A data type defines a kind of data that is represented by a variable

Java data types are case sensitive.
## Primitive Data Types

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Size (byte)</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>byte</td>
<td>1</td>
<td>-128 to 127</td>
</tr>
<tr>
<td>boolean</td>
<td>1</td>
<td>true or false</td>
</tr>
<tr>
<td>char</td>
<td>2 (Unicode)</td>
<td>A-Z, a-z, 0-9, etc.</td>
</tr>
<tr>
<td>short</td>
<td>2</td>
<td>-32768 to 32767</td>
</tr>
<tr>
<td>int</td>
<td>4</td>
<td>(about) -2 million to 2 million</td>
</tr>
<tr>
<td>float</td>
<td>4</td>
<td>-3.4E38 to 3.4E18</td>
</tr>
<tr>
<td>long</td>
<td>8</td>
<td>(about) -10E18 to 10E18</td>
</tr>
<tr>
<td>double</td>
<td>8</td>
<td>-1.7E308 to 1.7E308</td>
</tr>
</tbody>
</table>

- There are only eight primitive data types.
- New primitive data types cannot be created by a programmer.
Variables

- Variables are labels that describe a particular location in memory and associate it with a data type.

```java
static int numStudents = 2595;
```

- A declaration of a variable is where the program allow memory for the variable.
  - instance variables: declared without “static”
  - class variables: declared with “static”
  - local variables and Parameters
Operators

Operators are special symbols that perform specific operations on one, two, or three operands, and then return a result.

- assignment, arithmetic, and unary operators
  “=”; “+”; “*”; “++”; “!”
- equality, relational, and conditional operators
  “==”; “!”; “<=”; “||”; “?:”
- bitwise and bit shift operators
  “~”; “&”; “>>”
Expressions

- An expression is a construct made up of variables, operators, and method invocations.

```java
int sum = 5 + 9;
anArray[0] = 139;
Spirit mySpirit = new Spirit();
System.out.println("Hello world!");
```

- An expression can return other types of values as well.

- A compound expression can be constructed from various smaller ones.
A simple java Program

```java
//This program prints "Hello! Welcome to the Java World!"
package org.Welcome;

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

- Comments in Java are preceded in two types:
  - Line comments: Line comments are preceded by two slashes “/\*” in a line.
    ```java
    // This program prints "Hello! Welcome to the Java World!"
    ```
  - Paragraph comments: Paragraph comments are enclosed between “/\*” and “/\*/” in one or multiple lines.
    ```java
    /*This program prints "Hello! Welcome to the Java World!"*/
    ```
The second line of the program is the package name.

It specifies the package name, org.Welcome for the class Welcome.
The class is the essential Java construct.

A class is a template.

A program is defined by using one or more classes.
Reserved Words

- Reserved words (Keywords) are words that have a specific meaning to the compiler and cannot be used for other purposes (name of variables, classes, methods...) in the program.

- Examples:
  - When the compiler sees the word `class`, it understands the word after `class` is the name for the class.
  - `public, void, static` are also reserved words.
Modifiers

- Certain reserved words are called modifies that specify the properties of the data, methods, and classes and how they can be used.

- Examples:
  - A **public** datum, method, or class can be accessed by other programs.
  - **private**, **protected**, **static** and **void** are also modifiers.
A pair of braces in a program forms a block for a group components of a program.
 Statements

- A statement represents an action or a sequence of actions.
- A statement ends with a semicolon ";".

```
System.out.println("Hello! Welcome to the Java World!");
```
A method is a collection of statements that performs a sequence of operations to display a message on the console.

A method can be used even without fully understanding the details of how it works.

A method is used by invoking a statement with a string argument which is enclosed within parentheses.
The *main* method provides the control of the program flow.

The Java interpreter executes the application by invoking the *main* method.

```
public static void main(String[] args) {
}
```
Objects

- Objects are key to understand OO technology.
- An object is a structured block of data.
- An object may use many bytes of memory.
- An object stores its state in fields and exposes its behavior through methods.
- The data type of an object is its class.
- Creating an object is called instantiation.
The class is the essential Java construct.

A class is a blueprint or prototype from which objects are created.

A class is a description of a group of objects with similar properties and behaviors.

A class is a pattern for an object.

A class does not create any objects.
A class consists of
• a collection of fields, or variables, very much like the named fields of a struct.
• all the operations (called methods) that can be performed on those fields.
• can be instantiated.

A class describes objects and operations defined on those objects.
Constructors

- Classes should define one or more methods to create or construct instances of the class.
- Their name is the same as the class name.
- Constructors are differentiated by the number and types of their arguments.
- If none constructor is defined, a default one will be created.
- Constructors do not return anything.
public class circle {
    public static final double PI = 3.14159;
    public double r;    // instance field holds circle’s radius

    // The constructor method: initialize the radius field
    public void Circle(double r) { this.r = r; }  

    // Constructor to use if no arguments
    public void Circle() { r = 1.0; }

    // The instance methods: compute values based on radius
    public double circumference() { return 2 * PI * r; }
    public double area() { return PI * r*r; }
}
Inheritance

- Classes are arranged in a hierarchy.
- Inheritance enables to define a new class based on a class that already exists.
- A class that is derived from another class is called a subclass. The class from which the subclass is derived is called a superclass.
- A class inherits fields and methods from all its superclasses, whether direct or indirect.
- A subclass can override methods that it inherits, or it can hide fields or methods that it inherits.
public class PlaneCircle extends Circle {
    // Automatically inherit the fields and methods of Circle,
    public double cx, cy;

    // A new constructor method to initialize the new fields
    public PlaneCircle(double r, double x, double y) {
        super(r);
        this.cx = x;
        this.cy = y;
    }

    // The area() and circumference() methods are inherited from Circle
    // A new instance method that checks whether a point is inside the circle
    // It uses the inherited instance field r
    public boolean inside(double x, double y) {
        double dx = x - cx, dy = y - cy;
        double distance = Math.sqrt(dx*dx + dy*dy);
        return (distance < r);
    }
}
An interface defines a protocol of communication between two objects.

An interface declaration contains signatures, but no implementations, for a set of methods, and might also contain constant definitions.

A class that implements an interface must implement all the methods declared in the interface.

An interface name can be used anywhere a type can be used.
Example

```java
public interface GraphicObject {
    public double circumference();
    public double area();
}

public class circle implements GraphicObject {

    public static final double PI = 3.14159;
    public double r;

    // methods required to implement the GraphicObject interface
    public double circumference(){
        return 2 * PI * r;
    }

    public double area(){
        return PI * r*r;
    }
```
Encapsulation

- Encapsulation means hiding the details of an object’s internals from the other parts of a program. The object can be used only through its access methods, which are carefully written to keep the object consistent and secure.

- Encapsulation is designed to make an object look like a black box: The insides of the box are hidden from view.

- On the outside are some controls which are the only way that the user can use the box.
An abstract class is a class that is declared abstract.

Abstract classes cannot be instantiated, but they can be subclassed.

An abstract method is a method that is declared without an implementation.

Not everything defined in an abstract class needs to be abstract.

However, if a class includes even one abstract method, the class itself must be declared abstract.
Unlike interfaces, abstract classes can contain fields that are not static and final, and they can contain implemented methods.

Abstract classes provide a partial implementation, leaving it to subclasses to complete the implementation.

Abstract classes are most commonly subclassed to share pieces of implementation.

If an abstract class contains only abstract method declarations, it should be declared as an interface instead.
A package is a namespace for organizing a set of related classes and interfaces in a logical manner.

A package is a grouping of related types providing access protection and name space management.

Conceptually, packages can be thought as being similar to different folders on computer.

To create a package for a type, put a `package` statement as the first statement in the source file that contains the type (class, interface, enumeration, or annotation type).
Thank you!