Introduction to computer programming A
LAB10

贾艳红 Jana
Email:jiayh@mail.sustech.edu.cn

LAB OBJECTIVES

- Learn inheritance.
- Learn protected keyword.
- Learn to use enum types

knowledge points

Enumerated Types

- ✓ **Enumeration** means **a list of named constant**.**It is created using enum keyword**.
- ✓ **Enumerated type is type-safe, meaning that an attempt to assign a value other than one of the enumerated values or null will result in a compile error.**
- ✓ **Each enumeration constant is public, static and final by default.**
- ✓ you can use the following methods on an enumerated object:
  - public String name(); Returns a name of the value for the object
  - public int ordinal(); Returns the ordinal value associated with the enumerated value.
- ✓ In Java, enumeration defines a **class type** and have constructors, you **do not instantiate an enum using new.**
- ✓ The set of every element can be get by the method(**values()**),

```java
package sustech.cs102a.lab10;

public class DirectionTest {
    public static void main(String[] args) {
        Direction d = Direction.EAST;
        switch (d)
        {
            case EAST:
                System.out.println("Countries in the east: Japan, Korea");
                break;
            case SOUTH:
                System.out.println("Countries in the south: Australia, New Zealand");
                break;
            case NORTH:
                System.out.println("Countries in the north: Russia, Mongolia");
                break;
            case WEST:
                System.out.println("Countries in the west: US, Germany");
                break;
            default:
                System.out.println("error Direction!!");
        }
        System.out.println(d);
        System.out.println(d.toString());
        System.out.println(d.name());
        System.out.println(d.ordinal());
        Direction d1 = Direction.EAST;
        Direction d2 = Direction.SOUTH;
        System.out.println(d1.equals(d2));
        System.out.println(d1.compareTo(d2));

    }
}
enum Direction {
    NORTH, SOUTH, EAST, WEST
}
```

using switch statements

Can be defined directly without any new keyword

Defined by creating a list of enum variable

- ✓ An enumerated type is **a subclass of the Object class** and **the Comparable interface**, you can invoke the methods **equals, toString, and compareTo** from an enumerated object reference variable

Enumerated Types

- ✓ **In Java, enumeration defines a class type. so it can have constructors, methods and instance variables.**
  - **An private (by default) constructor** :While there is attribute defined for enum, there must be a private constructor with parameters for initialization on attributes
  - **There could be also get method** , **toString()** could also be override, other normal method could be nonstatic or static depends on the design.

```java
public enum Book {
    JHTP("Java: How to Program", "2012"),
    CHTP("C: How to Program"),
    CPPHTP("C++: How to Program", "2012"),
    VBHTP("Visual Basic: How to Program", "2011"),
    CSHARPHTP("Visual C#: How to Program");

    private String title;
    private final String year;

    private Book(String title, String year) {
        this.title = title;
        this.year = year;
    }

    private Book(String title) {
        this.title = title;
        this.year = "no info";
    }

    public String getTitle() {
        return title;
    }

    public String getYear() {
        return year;
    }
}
```

remeber to end with ;

```java
import java.util.EnumSet;

public class BookTest {
    public static void main(String[] args) {
        System.out.println("All books:");

        for (Book book : Book.values()) {
            System.out.printf("%-10s", book);
            System.out.printf("%-30s", book.getTitle());
            System.out.printf("%s\n", book.getYear());
        }

        System.out.println("\nDisplaying a range of enum constants:");

        for(Book book : EnumSet.range(Book.JHTP, Book.CPPHTP)) {
            System.out.printf("%-10s", book);
            System.out.printf("%-30s", book.getTitle());
            System.out.printf("%s\n", book.getYear());
        }
    }
}
```

```
All books:
JHTP      Java: How to Program          2012
CHTP      C: How to Program             no info
CPPHTP    C++: How to Program           2012
VBHTP     Visual Basic: How to Program  2011
CSHARPHTP Visual C#: How to Program     no info

Displaying a range of enum constants:
JHTP      Java: How to Program          2012
CHTP      C: How to Program             no info
CPPHTP    C++: How to Program           2012
```

Inheritance

Inheritance in Java is an important concept of OOP(Object Oriented Programming). it is the mechanism in java by which one class is allow to inherit the features(fields and methods) of another class.

Inheritance Syntax:

```
class Super {
    .....
    .....
}
class Sub extends Super {
    .....
    .....
}
```

superclass/base class/parentclass

subclass/derived class/child class

extends is the keyword used to inherit the properities of superclass

Why use inheritance?

- For Code Reusability.
- For Method Overriding (so runtime polymorphism can be achieved).

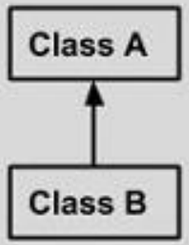
Inheritance

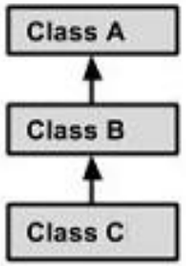
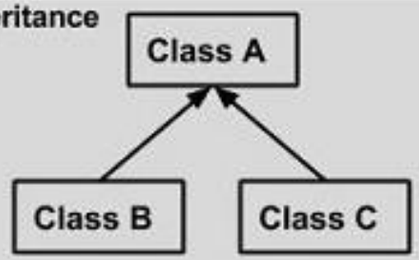
```java
package LAB10;


public abstract class Shape {
    protected double x;
    protected double y;
    protected ShapeColor color = ShapeColor.GRAY;
    private static int screenSize = 10;

    public Shape(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public double getX() {

        return x;
    }

    public void setX(double x) { this.x = x; }

    public double getY() { return y; }

    public void setY(double y) { this.y = y; }
```

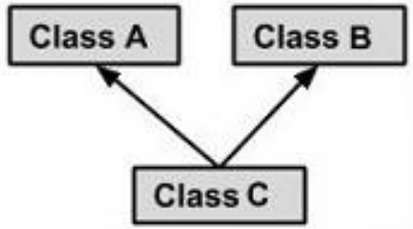
```java
import LAB10.ShapeColor;
import LAB10.StdDraw;
import LAB10.Shape;

public class Circle extends Shape {
    private double radius;
    static final int DEFAULT_RADIUS = 5;


    public Circle(double radius, double x, double y) {
        super(x,y);
        this.radius = radius;

    }

    public Circle(double radius) {
        super( x: 0, y: 0);
        this.radius = radius;

    }

    public Circle(double x, double y) {
        super(x,y);
        this.radius = DEFAULT_RADIUS;
```

The keyword **extends** tells the compiler the **Circle** class extends the **Shape** class, thus inheriting the methods **getX, setX, getY,setY, and toString etc.**

Types of Inheritance

Single Inheritance 	public class A {
	.....
	}
	public class B **extends** A {
	.....
	}

Multi Level Inheritance 	public class A { .....} public class B **extends** A {.....} public class C **extends** B {..... }
Hierarchical Inheritance 	public class A { .....} public class B **extends** A {.....} public class C **extends** A {..... }

Multiple Inheritance 	public class A { .....}
	public class B {.....}
	public class C **extends** A,B {
	.....
	} // Java does not support mutiple Inheritance

Why multiple inheritance is not supported?

Suppose, Class C inherits Class A and Class B. If there is the same method in both of these classes , it will create an ambiguity whenever we call the standard method using child class object

Inheritance

Super class vs sub class

- If a class **is not defined extends with a super class**, it inherit **java.lang.Object**
- **all the non-private data-filed and methods**（ except constructor ） of super class are inherited by the sub class from its super class.
- **sub class can also define its own data-filed and methods**
- **Superclass can only be one.**

Exercises

Complete the exercises in the **2021S-Java-A-Lab-10.pdf** and submit to the blackboard as required.

THANK YOU

贾艳红 Jana
Email:jiayh@mail.sustech.edu.cn