


Bộ môn Công nghệ Phần mềm
Viện CNTT & TT
Trường Đại học Bách Khoa Hà Nội

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG


Bài 04. Các kỹ thuật xây dựng lớp và sử dụng it ng



Mục tiêu bài học

- Nêu các bản chất, vai trò và bài tập sử dụng kỹ thuật lập trình hướng đối tượng, các kỹ thuật lập trình hướng đối tượng
- Thành viên lớp, thành viên lớp
- Hiểu về cách thức quản lý bộ nhớ và it ng trong Java
- Nắm vững cách thức truy cập tham số phương thức
- Biết cách sử dụng package, các lớp lập trình tích trong Java: Wrapper class, Math, System, String vs. StringBuffer


2



Nội dung

1. Các kỹ thuật lập trình hướng đối tượng
2. Thành viên lớp và thành viên lớp
3. Quản lý bộ nhớ trong Java
4. Truy cập tham số cho phương thức
5. Các lớp lập trình tích trong Java


3



Nội dung

1. Các kỹ thuật lập trình hướng đối tượng
2. Thành viên lớp và thành viên lớp
3. Quản lý bộ nhớ trong Java
4. Truy cập tham số cho phương thức
5. Các lớp lập trình tích trong Java


4



1.1. Các kỹ thuật lập trình hướng đối tượng

- Các kỹ thuật lập trình hướng đối tượng (Method Overloading)

5



1.1. Các kỹ thuật lập trình hướng đối tượng (2)

- Ví dụ 1:
 - Phương thức println() trong System.out.println()

6

1.1. Chương trình (3)

■ Ví dụ 2:

```
class MyDate {
    int year, month, day;
    public boolean setMonth(int m) { ...}
    public boolean setMonth(String s) { ...}
}

public class Test{
    public static void main(String args[]){
        MyDate d = new MyDate();
        d.setMonth(9);
        d.setMonth("September");
    }
}
```

7

Một số chú ý về chương trình

8

Thảo luận

```
void prt(String s) { System.out.println(s); }
void f2(short x) { prt("f3(short)"); }
void f2(int x) { prt("f3(int)"); }
void f2(long x) { prt("f5(long)"); }
void f2(float x) { prt("f5(float)"); }
```

■ Hãy suy nghĩ về các ví dụ sau:

- f2(5);
- char x='a'; f2(x);
- byte y=0; f2(y);
- float z = 0; f2(z);

■ Hãy suy nghĩ về các ví dụ sau:

9

1.2. Chương trình khác nhau

- Trong môi trường khác nhau các chương trình khác nhau

10

Ví dụ

```
public class BankAccount{
    private String owner;
    private double balance;
    public BankAccount(){owner = "noname";}
    public BankAccount(String o, double b){
        owner = o; balance = b;
    }
}

public class Test{
    public static void main(String args[]){
        BankAccount acc1 = new BankAccount();
        BankAccount acc2 =
            new BankAccount("Thuy", 100);
    }
}
```

11

1.3. Tài khóa này

12

Ví dụ

```
public class Ship {
    private double x=0.0, y=0.0;
    private double speed=1.0, direction=0.0;
    public String name;

    public Ship(String name) {
        this.name = name;
    }
    public Ship(String name, double x, double y) {
        this(name); this.x = x; this.y = y;
    }
    public Ship(String name, double x, double y,
        double speed, double direction) {
        this(name, x, y);
        this.speed = speed;
        this.direction = direction;
    }
}
//continue...
```

13

```
//(cont.)
private double degreeToRadian(double degrees) {
    return(degrees * Math.PI / 180.0);
}
public void move() {
    move(1);
}
public void move(int steps) {
    double angle = degreesToRadians(direction);
    x = x + (double)steps*speed*Math.cos(angle);
    y = y + (double)steps*speed*Math.sin(angle);
}
public void printLocation() {
    System.out.println(name + " is at ("
        + x + "," + y + ").");
}
} //end of Ship class
```

14

Nội dung

1. Chương trình cơ bản
2. Thành viên tĩnh và thành viên lớp
3. Quản lý bộ nhớ trong Java
4. Truy cập tham số cho chương trình
5. Một số lập trình ích trong Java

15

2.1. Thành viên static

```
public class MyDate {
    public static long getMillisSinceEpoch() {
        ...
    }
    ...
    long millis = MyDate.getMillisSinceEpoch();
}
```

16

Ví dụ lập JOptionPane trong javax.swing

Thuộc tính

Field Summary

static int	CANCEL_OPTION	Returns value from class method if CANCEL is chosen.
static int	CLOSED_OPTION	Returns value from class method if user closes window.
static int	DEFAULT_OPTION	Type used for showConfirmDialog.
static int	ERROR_MESSAGE	Used for error messages.
static int	WARNING_MESSAGE	Used for warning messages.
static int	YES_NO_CANCEL_OPTION	Type used for showConfirmDialog.
static int	YES_NO_OPTION	Type used for showConfirmDialog.
static int	YES_OPTION	Returns value from class method if YES is chosen.

```
static void showMessageDialog(Component parentComponent, Object message)
    Brings up an information-message dialog titled "Message".
static void showMessageDialog(Component parentComponent, Object message, String title, int messageType)
    Brings up a dialog that displays a message using a default icon determined by the messageType parameter.
static void showMessageDialog(Component parentComponent, Object message, String title, int messageType,
    Icon icon)
    Brings up a dialog that displays a message using a default icon determined by the messageType parameter.
```

17

Ví dụ sử dụng thuộc tính và phương thức static lập JOptionPane

```
JOptionPane.showMessageDialog(null, "Bạn đã thao tác  
lỗi", "Thông báo lỗi", JOptionPane.ERROR_MESSAGE);
```

```
JOptionPane.showConfirmDialog(null, "Bạn có chắc chắn  
muốn thoát?", "Hãy lựa chọn",  
JOptionPane.YES_NO_OPTION);
```

18

Ví dụ - sử dụng thuộc tính và phương thức static | JOptionPane (2)

- Object[] options = { "OK", "CANCEL" };
- JOptionPane.showOptionDialog(null, "Nhấn OK để tiếp tục", "Cảnh báo", JOptionPane.DEFAULT_OPTION, JOptionPane.WARNING_MESSAGE, null, options, options[0]);

19

2.1. Thành viên static (2)

- Thay đổi giá trị của một thành viên **static** trong một tệp tin class?

20

Ví dụ 1

```
class TestStatic{
    public static int iStatic;
    public int iNonStatic;
}

public class TestS {
    public static void main(String[] args) {
        TestStatic obj1 = new TestStatic();
        obj1.iStatic = 10; obj1.iNonStatic = 11;
        System.out.println(obj1.iStatic+" "+obj1.iNonStatic);
        TestStatic obj2 = new TestStatic();
        System.out.println(obj2.iStatic+" "+obj2.iNonStatic);
        obj2.iStatic = 12;
        System.out.println(obj1.iStatic+" "+obj1.iNonStatic);
    }
}
```

21

Ví dụ 2

```
public class Demo {
    int i = 0;
    void tang(){ i++; }
    public static void main(String[] args) {
        tang();
        System.out.println("Giá trị của i là" + i);
    }
}
```

22

2.2. Thành viên hằng

- Ví dụ :


```
final double PI = 3.141592653589793;
public final int VAL_THREE = 39;
private final int[] A = { 1, 2, 3, 4, 5, 6 };
```

23

Instance member vs. Class member

Thành viên instance Thành viên class

24

Nội dung

1. Chương trình thực
2. Thành viên T và thành viên I p
3. Quản lý bộ nhớ trong Java
4. Truy cập tham số cho phương thức
5. Một số lỗi phổ biến trong Java

25

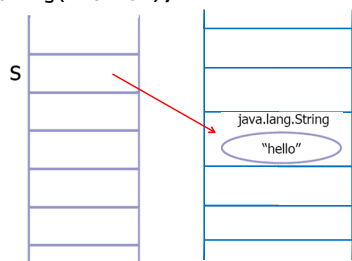
3. Quản lý bộ nhớ trong Java

- Java không sử dụng con tr

26

3.1. Bộ nhớ Heap

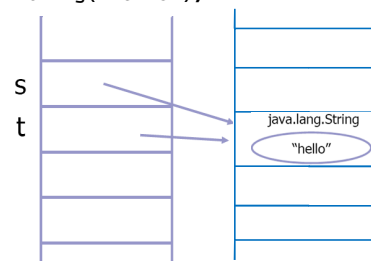
```
String s = new String("hello");
```



27

3.1. Bộ nhớ Heap

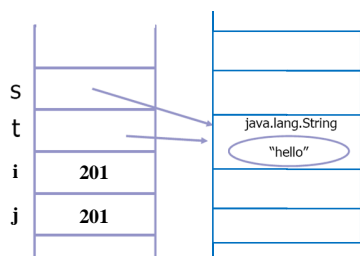
```
String s = new String("hello");
String t = s;
```



28

3.2. Bộ nhớ Stack

```
String s = new String("hello");
String t = s;
int i = 201;
int j = i;
```



29

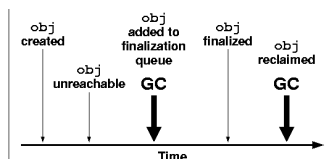
3.3. Bộ thu gom rác (Garbage Collector)

- Tiêu trình ch y ng m

30

Ph  ng th  c void finalize()

-   L  p n o c  ng c  ph  ng th  c finalize()



31

3.4. So s nh   t  ng

-   D   li   u th  ng v   i to n t   ==?

32

3.4. So s nh   t  ng (2)

-   i v   i c c   t  ng, to n t   == c    y ngh  a kh c
-   V   d   :

```
Employee a = new Employee(1);
Employee b = new Employee(1);
if (a==b)... // false
```

```
Employee a = new Employee(1);
Employee b = a;
if (a==b)... // true
```

33

3.4. So s nh   t  ng (3)

-   B   t k   i t  ng n o c  ng c  ph  ng th  c equals

34

V   d   == v  equals – L  p Integer

```
public class Equivalence {
    public static void main(String[] args) {
        Integer n1 = new Integer(47);
        Integer n2 = new Integer(47);
        System.out.println(n1 == n2);
        System.out.println(n1.equals(n2));
    }
}
```

35

V   d   3 – equals c   l p t   v   t

```
class Value {
    int i;
}

public class EqualsMethod2 {
    public static void main(String[] args) {
        Value v1 = new Value();
        Value v2 = new Value();
        v1.i = v2.i = 100;
        System.out.println(v1.equals(v2));
    }
}
```

36

Nội dung

1. Chương trình
2. Thành viên T và thành viên I p
3. Quản lý biến trong Java
4. Truy cập tham số cho phương thức
5. Một số lợi ích trong Java

37

4. Truy cập tham số cho phương thức

- Có thể sử dụng bất kỳ kiểu dữ liệu nào cho tham số của phương thức hoặc constructor

38

4. Truy cập tham số cho phương thức (2)

- Java: pass-by-value

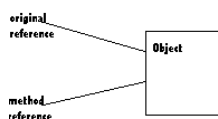
39

4.1. Ví dụ dữ liệu tham tr

- Các giá trị nguyên thủy không thay đổi khi truyền nhúng tham số

40

4.2. Ví dụ dữ liệu tham chiếu



41

Ví dụ

```

public class Point {
    private double x;
    private double y;
    public Point() { }
    public Point(double x, double y) {
        this.x = x; this.y = y;
    }
    public void setX(double x) { this.x = x; }
    public void setY(double y) { this.y = y; }
    public void printPoint() {
        System.out.println("X: " + x + " Y: " + y);
    }
}
  
```

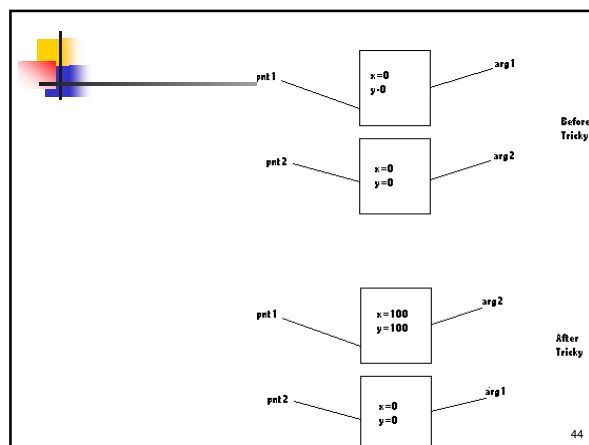
42

```

public class Test {
    public static void tricky(Point arg1, Point
arg2) {
        arg1.setX(100); arg1.setY(100);
        Point temp = arg1;
        arg1 = arg2; arg2 = temp;
    }
    public static void main(String [] args) {
        Point pnt1 = new Point(0,0);
        Point pnt2 = new Point(0,0);
        pnt1.printPoint(); pnt2.printPoint();
        System.out.println(); tricky(pnt1, pnt2);
        pnt1.printPoint(); pnt2.printPoint();
    }
}

```

43



44

4.3. Truy n s l ng tham s tùy ý

- c g i là varargs.
- Ví d :
 - `System.out.printf ("%s: %d, %s\n", name, idnum, address);`
 - `System.out.printf ("%s: %d, %s, %s, %s\n", name, idnum, address, phone, email);`

45

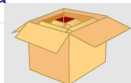
N i dung

1. Ch ng ph ng th c
2. Thành viên T và thành viên l p
3. Qu n lý b nh trong Java
4. Truy n tham s cho ph ng th c
5. M t s l p t i n ích trong Java

46

5.1. Package trong Java

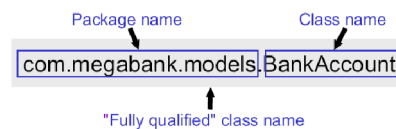
- Package gi ng nh th m c



47

5.1. Package trong Java (2)

- Tên y c a l p:



48

a. Tham chi u gi a các l p

■ Ví d :

```
public class HelloNameDialog{
    public static void main(String[] args){
        String result;
        result = javax.swing.JOptionPane.showInputDialog
            ("Hay nhap ten ban:");
        javax.swing.JOptionPane.showMessageDialog(null,
            "Xin chao "+ result + "!");
    }
}
```

49

a. Tham chi u gi a các l p (2)

■ L nh **import**:

- S d ng l nh **import** khai báo các package ho c các l p khi s d ng không c n nêu tên y .

50

b. Các package trong Java

• java.applet	• javax.rmi
• java.awt	• javax.security
• java.beans	• javax.sound
• java.io	• javax.sql
• java.lang	• javax.swing
• java.math	• javax.transaction
• java.net	• javax.xml
• java.nio	• org.ietf.jgss
• java.rmi	• org.omg.CORBA
• java.security	• org.omg.CosNaming
• java.sql	• org.omg.Dynamic
• java.text	• org.omg.IOP
• java.util	• org.omg.Messaging
• javax.accessibility	• org.omg.PortableInterceptor
• javax.crypto	• org.omg.PortableServer
• javax.imageio	• org.omg.SendingContext
• javax.naming	• org.omg.stub.java.rmi
• javax.net	• org.w3c.dom
• javax.print	• org.xml

51

b. Các package trong Java (2)

■ Các package c b n trong Java

- java.lang
- java.util
- java.io

52

b. Các package trong Java (3)

■ Các package c b n trong Java

- java.math
- java.sql
- javax.swing

53

5.2. Các l p bao (Wrapper class)

- M i ki u d li u nguyên th y có m t l p t ng ng g i là l p bao:

54

5.2. Các lớp bao (2)

Primitive Type	Wrapper Class
boolean	Boolean
byte	Byte
char	Character
double	Double
float	Float
int	Integer
long	Long
short	Short

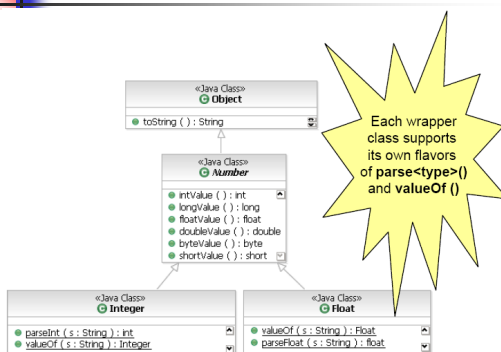
55

a. Chuyển đổi kiểu dữ liệu

- Sử dụng `toString()`
- Sử dụng `<type>Value()`
- Sử dụng `parse<type>()` và `valueOf()`

56

a. Chuyển đổi kiểu dữ liệu (2)



57

b. Các hằng số

- Boolean**
 - Boolean FALSE
 - Boolean TRUE
- Byte**
 - byte MIN_VALUE
 - byte MAX_VALUE
- Character**
 - int MAX_RADIX
 - char MAX_VALUE
 - int MIN_RADIX
 - char MIN_VALUE
 - Unicode classification constants
- Double**
 - double MAX_VALUE
 - double MIN_VALUE
 - double NaN
 - double NEGATIVE_INFINITY
 - double POSITIVE_INFINITY
- Float**
 - float MAX_VALUE
 - float MIN_VALUE
 - float NaN
 - float NEGATIVE_INFINITY
 - float POSITIVE_INFINITY
- Integer**
 - int MIN_VALUE
 - int MAX_VALUE
- Long**
 - long MIN_VALUE
 - long MAX_VALUE
- Short**
 - short MIN_VALUE
 - short MAX_VALUE

58

Ví dụ

```
double d = (new Integer(Integer.MAX_VALUE)).
           doubleValue();
System.out.println(d);

String input = "test 1-2-3";
int output = 0;
for (int index=0; index<input.length(); index++) {
    char c = input.charAt(index);
    if (Character.isDigit(c))
        output = output * 10 + Character.digit(c, 10);
}
System.out.println(output);
```

59

5.3. Xâu (String)

- Kiểu String là một lớp và không phải là kiểu dữ liệu nguyên thủy

60

a. Ghép xâu

- Toán tử +
- Các kiểu dữ liệu cơ bản sử dụng trong `l i g i` `println()` chuyển nội dung sang kiểu String

61

b. Các phương thức của xâu

```
String name = "Joe Smith";
name.toLowerCase();
name.toUpperCase();
"Joe Smith ".trim();
"Joe Smith".indexOf('e');
"Joe Smith".length();
"Joe Smith".charAt(5);
"Joe Smith".substring(5);
"Joe Smith".substring(2,5);
```

62

c. So sánh hai xâu

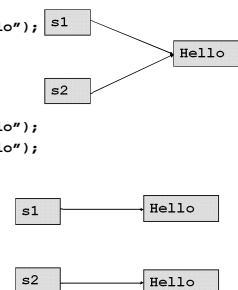
- `oneString.equals(anotherString)`
- `oneString.equalsIgnoreCase(anotherString)`
- So sánh `oneString == anotherString` gây nhầm lẫn

63

c. So sánh hai xâu (2)

```
String s1 = new String("Hello");
String s2 = s1;

String s1 = new String("Hello");
String s2 = new String("Hello");
```



64

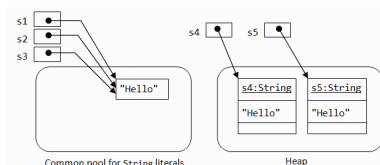
d. Đặc điểm của String

- Khởi tạo String theo 2 cách:
 - Gán 1 giá trị literal
 - Dùng toán tử `new` (Không khuyến khích dùng)
- Ví dụ:
 - `String str1 = "Java is Hot";`
 - `String str2 = new String("I'm cool");`

65

String Literal vs. String Object

- `String s1 = "Hello";`
- `String s2 = "Hello";`
- `String s3 = s1;`
- `String s4 = new String("Hello");`
- `String s5 = new String("Hello");`



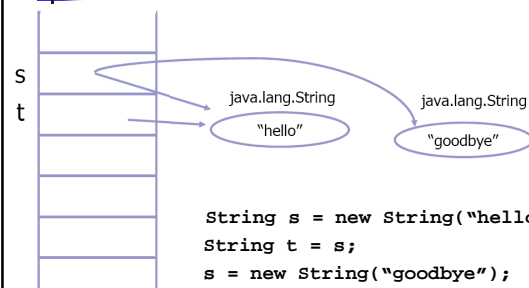
66

5.4. StringBuffer

- String là kiểu bất biến:
- StringBuffer là kiểu biến đổi:

67

5.4. StringBuffer (2)



68

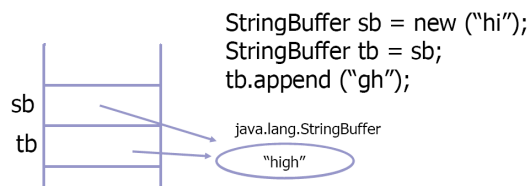
5.4. StringBuffer (3)

- StringBuffer:**
 - Khi nào dùng?

69

5.4. StringBuffer (4)

- Tính biến đổi



70

5.4. StringBuffer (5)

```
StringBuffer buffer = new StringBuffer(15);
buffer.append("This is ");
buffer.append("String");
buffer.insert(7," a");
buffer.append('.');
System.out.println(buffer.length());
System.out.println(buffer.capacity());
String output = buffer.toString();
System.out.println(output);
```

71

5.5. Lớp Math

- java.lang.Math cung cấp các thành phần static:

```

«Java Class»
Math
+ Math ()
+ abs ()
+ acos ()
+ asin ()
+ atan ()
+ atan2 ()
+ toRadians ()
+ toDegrees ()
+ exp ()
+ log ()
+ log10 ()
+ pow ()
+ round ()
+ rint ()
+ random ()
+ random ()
+ abs ()
+ abs ()
+ abs ()
+ abs ()
+ max ()
+ max ()
+ max ()
+ max ()
+ min ()
+ min ()
+ min ()
+ min ()
+ <clinit> ()

```

72

5.5. L p Math (2)

■ Ví d :

$$e^{\sqrt{2\pi}}$$

```
Math.pow(Math.E,
    Math.sqrt(2.0*Math.PI))
```

Ho c:

```
Math.exp(Math.sqrt(2.0*Math.PI))
```



5.6. L p System

- java.lang.System ch a nhi u hàm ti n ích h u d ng

5.6. L p System (2)

- `currentTimeMillis()`:
- `exit()`:
- `gc()`:
- Các ph ng th c liên quan n thu c tính c a h th ng:

```
System.out.println(System.currentTimeMillis());
```

5.6. L p System (3)

```
import java.util.Properties;
public class PropertiesTest {
    public static void main(String[] args) {
        System.out.println(
            System.getProperty("path.separator"));
        System.out.println(
            System.getProperty("file.separator"));
        System.out.println(
            System.getProperty("java.class.path"));
        System.out.println(
            System.getProperty("os.name"));
        System.out.println(
            System.getProperty("os.version"));
        System.out.println(System.getProperty("user.dir"));
        System.out.println(System.getProperty("user.home"));
        System.out.println(System.getProperty("user.name"));
    }
}
```