Bài 6. Kiểm soát truy cập, phạm vi biến lớp, tổ chức gói, Java API

THCS 4: Lập trình cơ bản với Java

Đỗ Thanh Hà, Nguyễn Thị Minh Huyền

Bộ môn Tin học
Khoa Toán - Cơ - Tin học
Trường Đại học Khoa học Tự nhiên

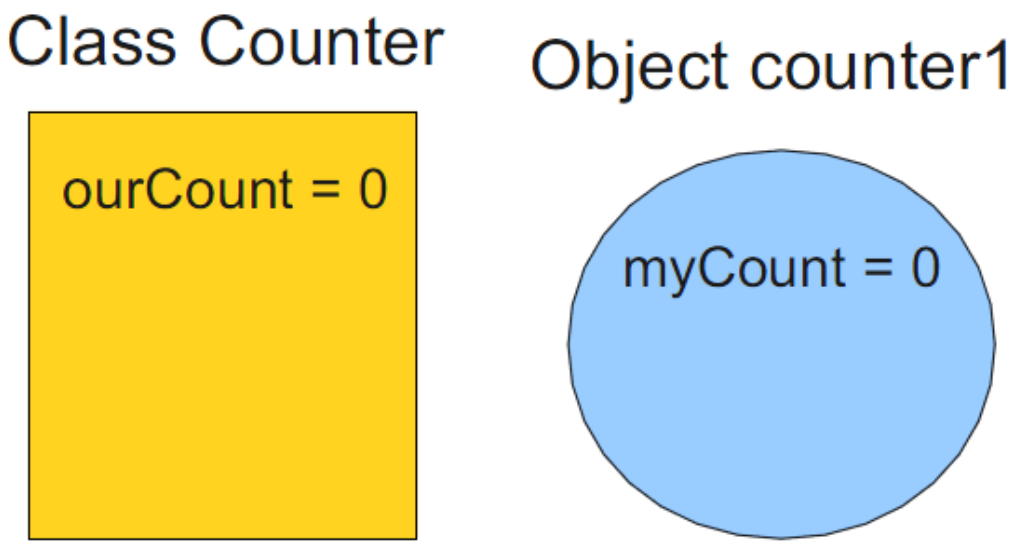
Nội dung

Nội dung

Ôn tập

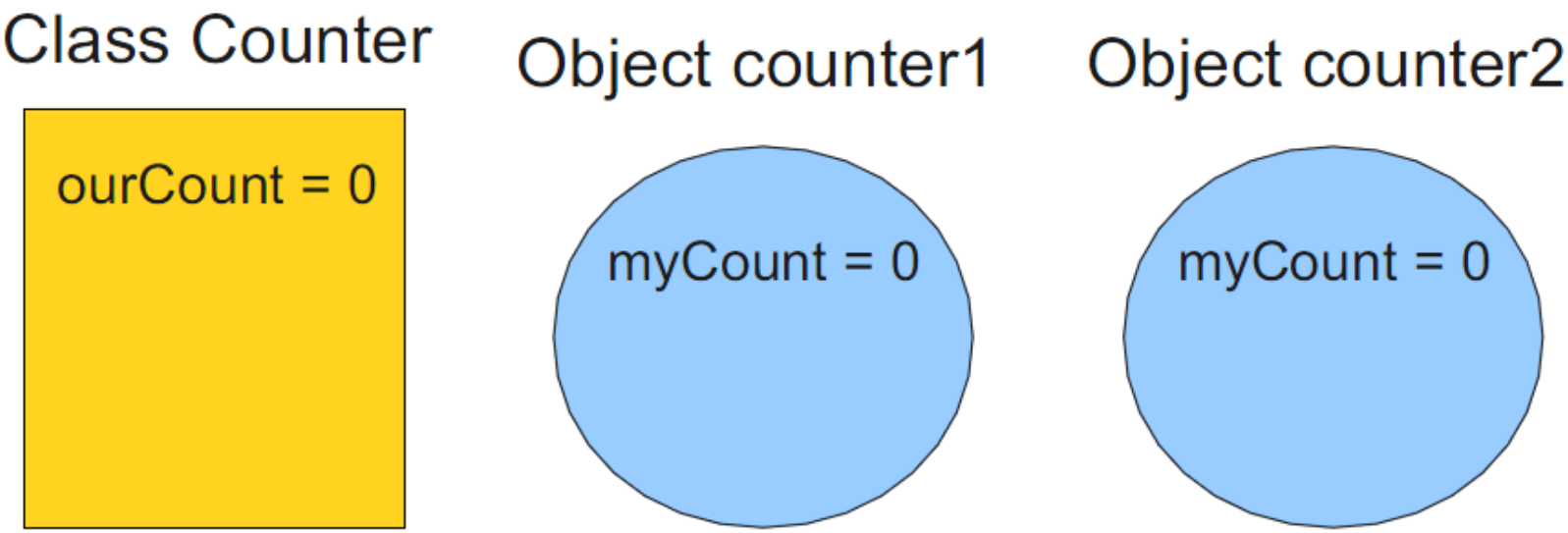
```
public class Counter{
   int myCount = 0;
   static int ourCount = 0;
   void increment(){
      myCount++;
      ourCount++;
   }
   public static void main(String arg[]){
      Counter counter1 = new Counter();
      Counter counter2 = new Counter();
      counter1.increment();
      counter1.increment();
      counter2.increment();
      System.out.println("counter 1: " +
          counter1.myCount + " " + counter1.ourCount);
      System.out.println("counter 2: " +
          counter2.myCount + " " + counter2.ourCount);
  }
```

Ôn tập



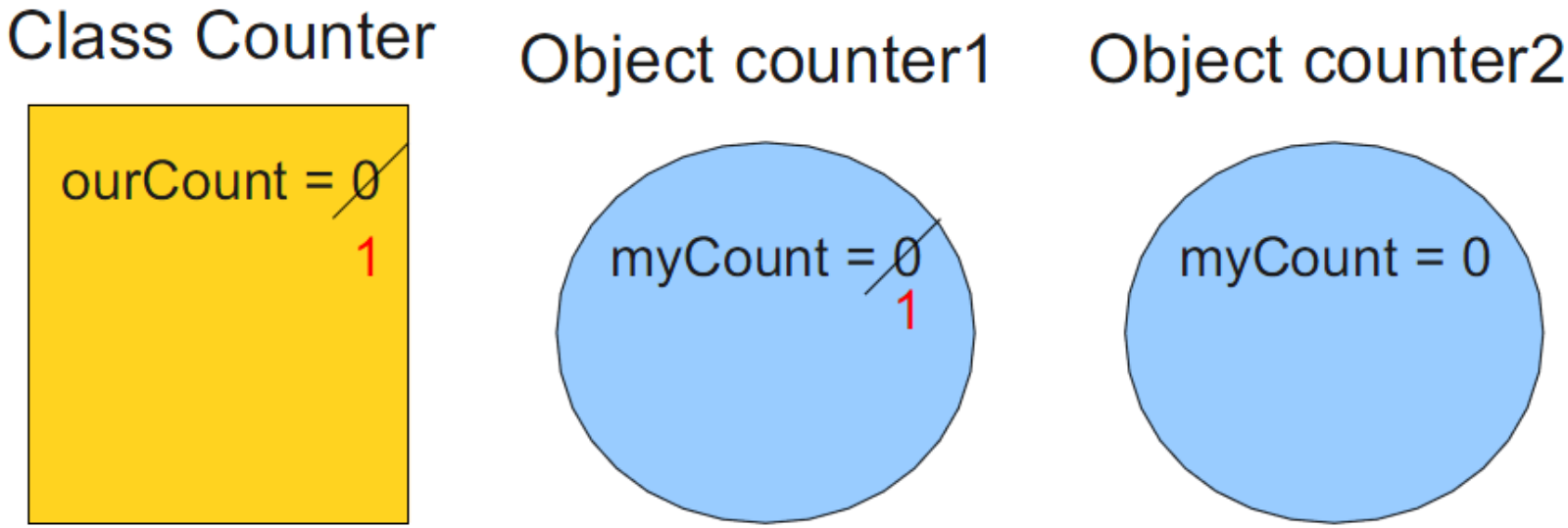

```
Counter counter1 = new Counter();
```

Ôn tập



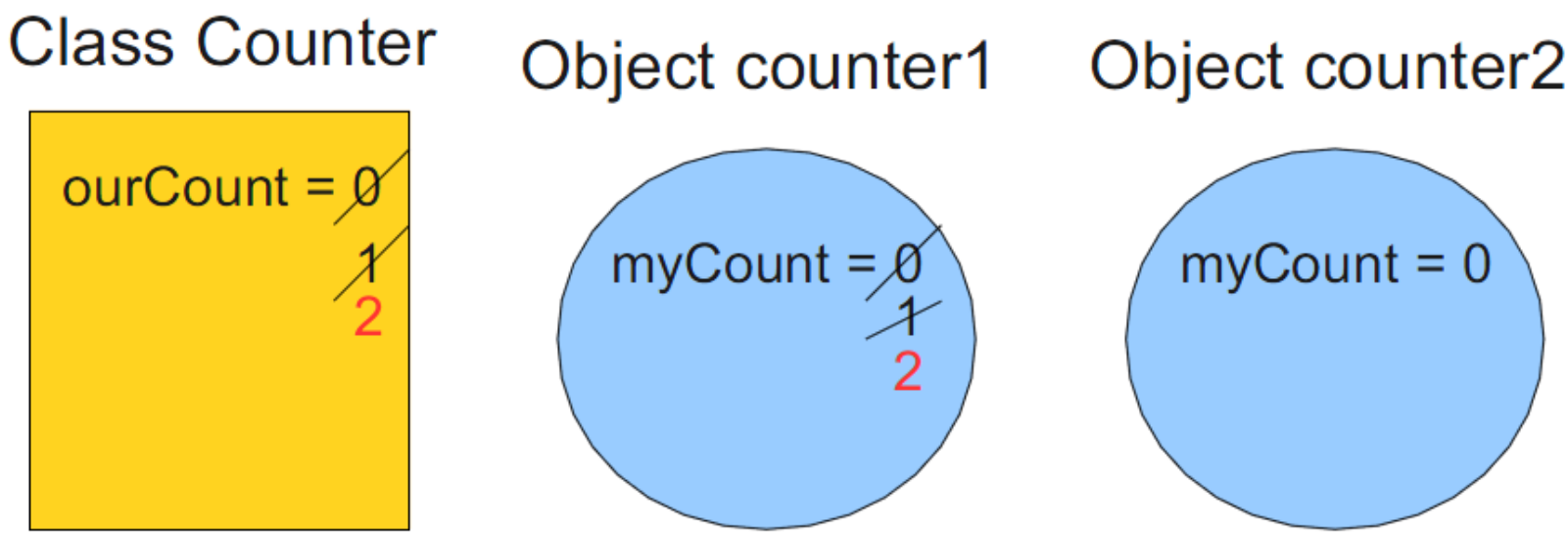

```
Counter counter1 = new Counter();
Counter counter2 = new Counter();
```

Ôn tập



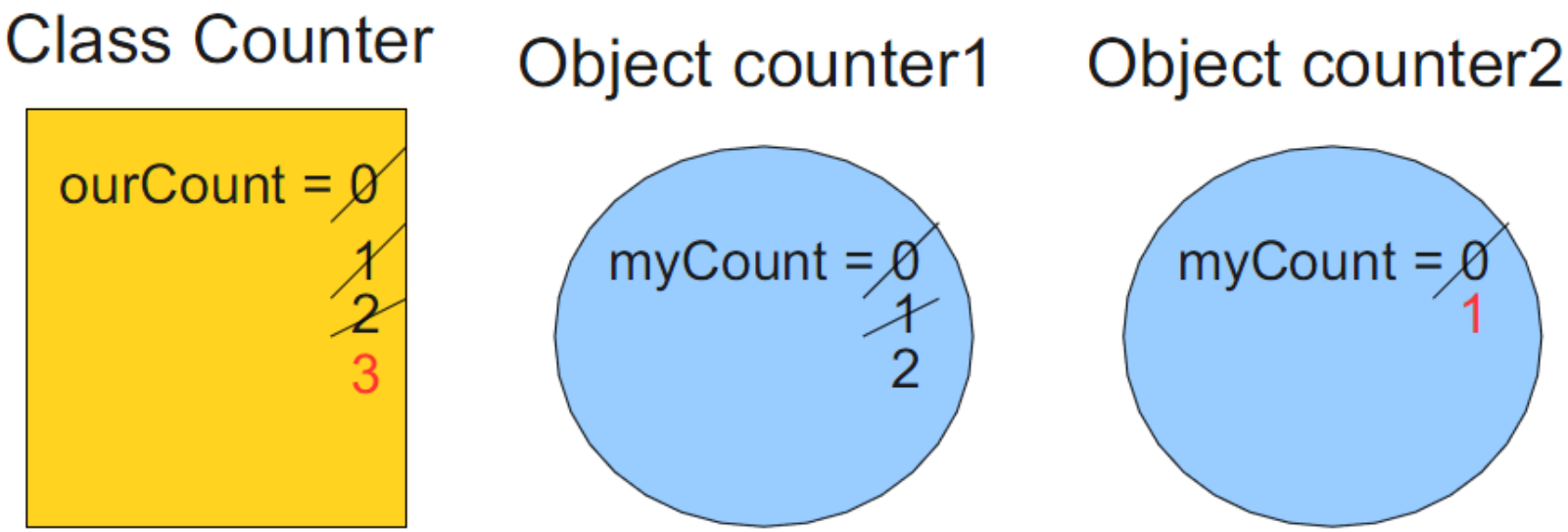

```
Counter counter1 = new Counter();
Counter counter2 = new Counter();
counter1.increment();
```

Ôn tập




```
Counter counter1 = new Counter();
Counter counter2 = new Counter();
counter1.increment();
counter1.increment();
```

Ôn tập




```
Counter counter1 = new Counter();
Counter counter2 = new Counter();
counter1.increment();
counter1.increment();
counter2.increment();
```

Nội dung

Kiểm soát truy cập (1)

```
public class CreditCard{
    String cardNumber;
    double expenses;

    void charge(double amount){
        expenses += amount;
    }

    String getCardNumber(String password){
        if(password.equals("SECRET!3*!")){
            return cardNumber;
        }
        return "robber";
    }
}
```

Kiểm soát truy cập (2)

```
public class Malicious{
    public static void main(String args[]){
        maliciousMethod(new CreditCard());
    }

    static void maliciousMethod(CreditCard card){
        card.expenses = 0;
        System.out.println(card.cardNumber);
    }
}
```

public và **private**

- **public**
  - Tất cả các lớp khác có thể truy cập trực tiếp tới thuộc tính của đối tượng hay gọi phương thức của đối tượng
- **private**
  - Chỉ có thể truy cập thuộc tính hay phương thức từ trong chính lớp đó

Kiểm soát truy cập áp dụng cho các
thuộc tính và **phương thức**

Kiểm soát truy cập (3)

```
public class CreditCard{
    private String cardNumber;
    private double expenses;

    public void charge(double amount){
       expenses +=amount;
    }

    public String getCardNumber(String password){
       if(password.equals("SECRET!3*!")){
           return cardNumber;
       }
       return "robber";
   }
}
```

Vì sao kiểm soát truy cập?

- Bảo vệ các thông tin riêng
- Làm rõ cách sử dụng lớp
- Tách biệt cài đặt và giao diện lớp

Nội dung

Phạm vi biến trong lớp

- Phạm vi biến trong phương thức (nhắc lại):
  - từ vị trí khai báo cho tới dấu đóng ngoặc } tương ứng với dấu mở ngoặc { gần nhất xuất hiện trước nó
- Phạm vi biến ở mức lớp
  - trong toàn lớp
  - Ví dụ:

```
public class Example{
    private int memberVariable;
    public void setVariable(int value){
       memberVariable = value;
    }
}
```

Từ khóa **this**

- Tên biến trong phương thức có thể trùng với tên biến ở mức lớp
  - Khi đó từ khóa this dùng để phân biệt biến lớp với biến của phương thức
  - **this**: "đối tượng này"
  - Ví dụ:

```
public class Example{
    private int memberVariable;
    public void setVariable(int memberVariable){
       this.memberVariable = memberVariable;
    }
}
```

Nội dung

Tổ chức gói (*package*)

- Mục đích: tổ chức các lớp sao cho việc tra cứu, sử dụng được thuận tiện
  - Mỗi lớp thuộc vào một gói (package) nào đó
  - Các lớp trong cùng một gói phục vụ mục đích tương tự nhau
  - Cho phép phân biệt các lớp khác nhau nhưng trùng tên
- Các package đơn giản là các thư mục
- Trong chương trình nếu sử dụng lớp thuộc các gói khác (*thư mục khác*) thì cần phải khai báo (*import*)

Định nghĩa gói

- Định nghĩa gói của lớp ở đầu tệp

```
package path.to.package.foo;
class Foo{
    // ..... do somthings
}
```

- Tệp Foo.java chứa lớp này phải nằm trong thư mục con path\to\package\foo của thư mục làm việc hiện thời
- Khai báo sử dụng một lớp cụ thể trong gói hay tất cả các lớp trong gói

```
import path.to.package.foo.Foo;
import path.to.package.foo.*;
```

```
package parenttools;

public class BabyFood {

}
```

```
package parenttools;

public class Baby {

}
```

```
package adult;

import parenttools.Baby;
import parenttools.BabyFood;

public class Parent {
 public static void main(String[] args) {
    Baby baby = new Baby();
    baby.feed(new BabyFood());
 }
}
```

Lưu ý khi sử dụng gói

- Tất cả các lớp "*nhìn thấy*" các lớp trong cùng gói (cùng thư mục) - không cần khai báo (*import*) các lớp này khi sử dụng
- Tất cả các lớp "*nhìn thấy*" các lớp trong gói java.lang
  - java.lang.System
  - java.lang.String
  - ....
- Mỗi lớp chỉ sử dụng được các lớp trong gói khác nếu các lớp này đã được khai báo **public**

Kiểm soát truy cập

	class	package	subclass	world
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

Ví dụ

- Lớp demo.lect6.PackageExample và lớp PackageUsage
- Sử dụng gói dùng import
- Sử dụng gói không dùng import

Nội dung

Java API

- Thư viện Java có rất nhiều gói/lớp
- Sử dụng lại các lớp có sẵn giúp người lập trình tiết kiệm nhiều công sức
- http://docs.oracle.com/javase/N/docs/api/

(N = *phiên bản Java*)

Tra cứu một số phương thức của lớp String?

Kiểu dữ liệu ArrayList

- Mảng: kích thước cố định sau khi khởi tạo
  - bất tiện nếu số phần tử không được biết trước
- ArrayList: mảng có kích thước thay đổi
- Sử dụng ArrayList:
  - Lấy kích thước mảng (phương thức **size**)
  - Thêm phần tử (phương thức **add**)
  - Đọc/thay đổi giá trị phần tử (phương thức **get/set**)
  - Xóa phần tử (phương thức **remove**)
  - Lặp trên các phần tử
  - Ví dụ: ArrayListExample.java