LẬP TRÌNH JAVA

Bài 4: Lớp và Đối tượng

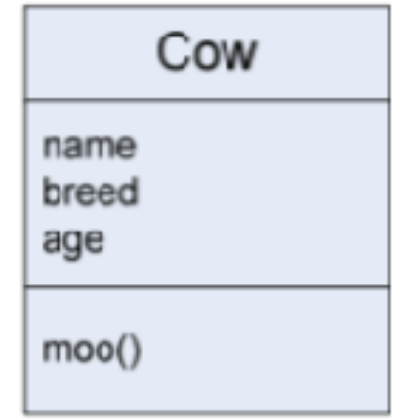
Khái niệm

- **Đối tượng** là một thực thể có trạng thái và hành vi, ví dụ như bàn, ghế, xe con, mèo
- **Lớp** là một mẩu mô tả trạng thái, hành động của đối tượng đó.
- Lớp là khuôn mẫu để từ đó tạo ra các thực thể.
- Có hai loại thông tin quan trọng về mỗi đối tượng:
  - Những thông tin mà đối tượng đó biết (thuộc tính – attribute)
  - Những việc mà đối tượng đó làm (hành vi – method)

Tạo và sử dụng đối tượng

```
class Cow {
  String name;
  String breed;
  int age;

  void moo() {
    System.out.println("Moo...!");
  }
}
```

một phương thức




```
public class CowTestDrive {
  public static void main (String[] args) {
    Cow c = new Cow();  // make a Cow object
    c.age = 2; // set the age of the Cow
    c.moo();   // call its moo() method
  }
}
```

Những điểm quan trọng:

- Tất cả mã Java đều nằm trong một lớp nào đó.
- Một lớp đặc tả cách tạo một đối tượng thuộc lớp đó. Một lớp giống như một bản thiết kế
- Một đối tượng có thể tự lo cho bản thân, ta không phải cần biết hay quan tâm một đối tượng làm việc đó như thế nào.
- Một đối tượng biết về một số thứ và có thể làm một số việc.
- Những gì một đối tượng biết về chính nó được gọi là các biến thực thể (thuộc tính) của nó. Chúng đại diện cho trạng thái của đối tượng đó.
- Những gì một đối tượng có thể làm được gọi là các phương thức. Chúng đại diện cho hành vi của đối tượng đó.
- Khi viết một lớp, ta có thể muốn viết một lớp khác để test. Tại đó ta tạo các đối tượng thuộc lớp kia và thử nghiệm với chúng.
- Tại thời gian chạy, một chương trình Java chính là một nhóm các đối tượng đang "nói chuyện" với nhau.

Biến thực thể, biến địa phương

Biến thực thể là biến được khai báo bên trong một lớp nhưng không nằm trong một phương thức nào.

Biến địa phương là biến được khai báo bên trong một phương thức.

Biến địa phương phải được khởi tạo trước khi sử dụng.

Biến thực thể, biến địa phương

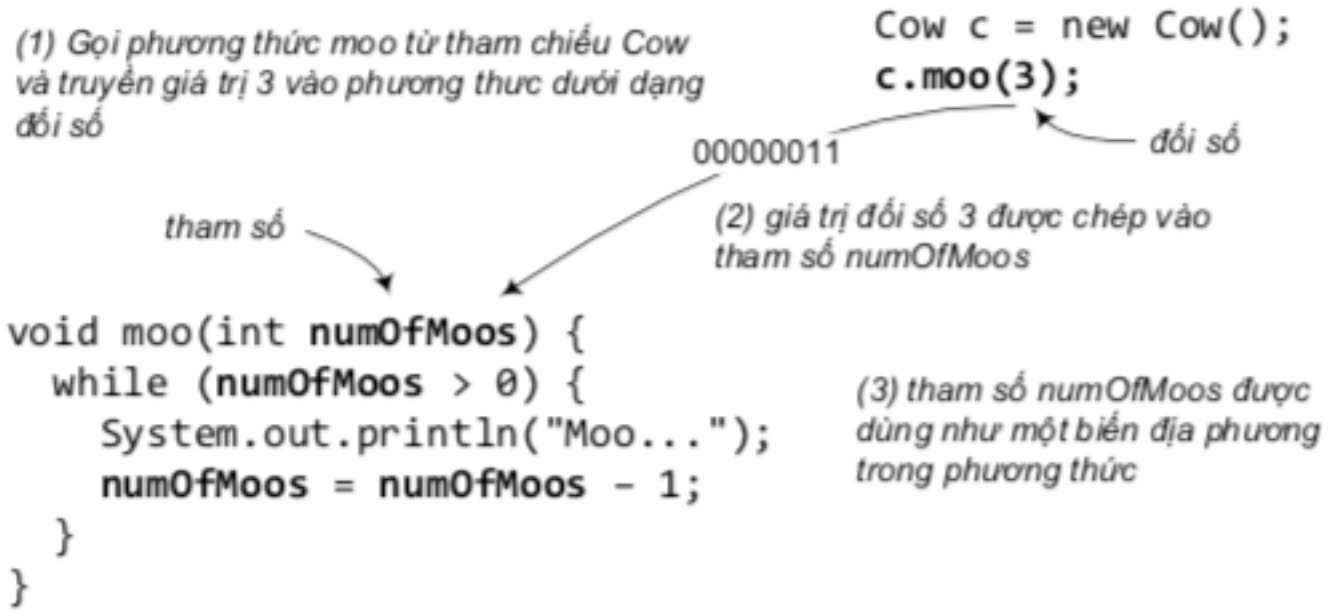
```
class Foo {
  int a = 1;
  int b;

  public int add() {
    int sum = a + b;
    return sum;
  }
  public int addThatWontCompile() {
    int dummy;
    int sum = a + dummy;
    return sum;
  }
}
```

a là biến thực thể chưa được khởi tạo nhưng đã có giá trị mặc định

lỗi biên dịch do dùng biến địa phương dummy chưa được khởi tạo

Tham số và giá trị trả về

Tham số và đối số



Tham số và giá trị trả về

Giá trị trả về

Tham số và giá trị trả về

Công dụng từ khóa this.

* Truy cập thuộc tính của chính đối tượng.

```
class MyInteger {
  private int value;
  public boolean greaterThan(MyInteger other) {
    return (this.value > other.value);
  }
  public boolean lessThan(MyInteger other) {
    return (other.greaterThan(this));
  }
  public MyInteger increment() {
    value++;
    return this;
  }
}
```

this dùng để truy nhập biến thực thể

this làm đối số

this làm giá trị trả về

```
MyInteger counter = new MyInteger();
counter.increment().increment(); // tăng hai lần
```

Đóng gói và truy cập

Các phương thức đọc dữ liệu của đối tượng và trả về dữ liệu đọc được. Chúng thường được đặt tên là **getDữLiệuGìĐó**, nên còn được gọi là các phương thức **get**.

Các phương thức ghi dữ liệu vào các biến thực thể của đối tượng, chúng nhận dữ liệu mới qua các tham số rồi ghi vào các biến liên quan. Chúng thường được đặt tên là **setDữLiệuGìĐó**, nên còn được gọi là các phương thức **set**

Đóng gói và truy cập

```
class Cow {
  String name;
  int age;

  void setName(String aName) {
    name = aName;
  }
  String getName() {
    return name;
  }
  void setAge(int anAge) {
    age = anAge;
  }
  int getAge() {
    return age;
  }
}
```

Có thể chỉnh sửa biến **age** không cần dùng phương thức **getAge, setAge** được không?

-> Được

Cow cow = new Cow();

int a = cow.age;

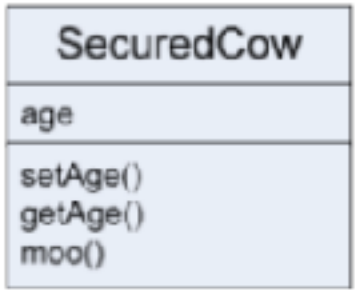
cow.age = 10

Đóng gói đối tượng

Hạn chế truy cập vào thuộc tính bằng các **Access modifier**.

- Public
- Private
- Protected
- Default

```
class SecuredCow {
  private int age;

  public void setAge(int a) {
    if (a>0) {
      age = a;
    }
  }
  public int getAge() {
    return age;
  }
  void moo() {
    if (age <= 5) {
      System.out.println("Mooooooo...");
    } else {
      System.out.println("Moo.");
    }
  }
}
```



Bài tập

1. Viết class Employee chứa ba mẩu thông tin dưới dạng các thành viên dữ liệu: tên (first name, kiểu String), họ (last name, kiểu String) và lương tháng (salary, kiểu double). Class Employee cần có một hàm khởi tạo có nhiệm vụ khởi tạo ba thành viên dữ liệu này. Hãy viết một hàm set và một hàm get cho mỗi thành viên dữ liệu. Nếu lương tháng có giá trị âm thì hãy gán cho nó giá trị 0.0. Viết một chương trình thử nghiệm EmployeeTest để chạy thử các tính năng của class Employee. Tạo hai đối tượng Employee và in ra màn hình tổng lượng hàng năm của mỗi người. Sau đó cho tăng lương cho mỗi người thêm 10% và hiển thị lại lương của họ theo năm.

Bài tập

2. Tạo một lớp có tên Invoice (hóa đơn) mà một cửa hàng có thể dùng để biểu diễn một hóa đơn cho một món hàng được bán ra tại cửa hàng. Mỗi đối tượng Invoice cần có 4 thông tin chứa trong các thành viên dữ liệu: số hiệu của mặt hàng (partNumber kiểu String), miêu tả mặt hàng (partDescription kiểu String), số lượng bán ra (quantity kiểu int) và đơn giá (unitPrice kiểu double). Lớp Invoice cần có một hàm khởi tạo có nhiệm vụ khởi tạo 4 thành viên dữ liệu đó. Hãy viết một phương thức set và một phương thức get cho mỗi thành viên dữ liệu. Ngoài ra, hãy viết một phương thức có tên getInvoiceAmount với nhiệm vụ tính tiền hóa đơn (nghĩa là số lượng nhân với đơn giá), rồi trả về giá trị hóa đơn dưới dạng một giá trị kiểu double. Nếu số lượng không phải số dương thì cần gán cho nó giá trị 0. Nếu đơn giá có giá trị âm, nó cũng cần được gán giá trị 0.0. Viết một ứng dụng thử nghiệm tên là InvoiceTest để chạy thử các tính năng của class Invoice.

Bài tập

3. Bài toán quản lý thông tin 3 học sinh bao gồm những thông tin sau: Họ tên, ngày sinh, điểm toán, lý hóa, anh. Người quản lý sẽ quản lý về thông tin các học sinh trên, hiển thị danh sách sinh viên theo điểm tăng dần của điểm trung bình. Ngoài ra khi cần thiết, sẽ tiến hành tìm kiếm theo họ tên hoặc các điểm toán, lý, hóa, anh để hiển thị ra các thông tin học sinh cần tìm.
Viết chương trình giải quyết bài toán trên.