**Khái niệm**

* Trong thế giới thực ta dễ dàng nhận thấy rất nhiều những thực thể hay đối tượng có các đặc điểm giống hệt nhau. Ví dụ như những chiếc iPhone 12 Pro Max, những chiếc xe Tesla model X, những chiếc Macbook Pro phiên bản 202x, những chiếc laptop ASUS dòng XYZ nào đó… Tất cả các sản phẩm cùng phiên bản đó được tạo ra từ cùng một bản thiết kế. Do đó có các chức năng, đặc điểm và giá thành tương tự nhau.
* Một đối tượng thường có các đặc điểm nhận biết và hành động cụ thể.
* Từ ý tưởng này, người ta cũng muốn tạo ra một chuẩn chung, bản mẫu chung để mô tả thông tin các đối tượng giống nhau trong ngôn ngữ lập trình. Để làm được điều này, ta sử dụng class.
* ***Một lớp là một bản mô tả cụ thể(có thể coi như bản vẽ, mẫu thiết kế, mô hình hóa) của một đối tượng trong ngôn ngữ lập trình.***
* ***Hoặc: một lớp là một bản mẫu thiết kế chi tiết mà từ đó các đối tượng riêng lẻ của nó được tạo ra.***
* Java là ngôn ngữ lập trình hướng đối tượng, đặc trưng của nó là tất cả các thành phần được xây dựng trên các lớp.
* Các đối tượng có thể là đối tượng hữu hình, cụ thể: cái xe máy, ô tô, smartphone… hoặc cũng có thể là các đối tượng trừu tượng: đơn hàng, giỏ hàng, tài khoản facebook,…
* Bất cứ thứ gì có thể dùng danh từ để đặt tên cho nó thì đều có thể đem nó vào làm đối tượng trong ngôn ngữ lập trình.

**Mục đích sử dụng**

* Vậy, mục đích của class là để mô hình hóa một đối tượng từ thế giới thực vào thế giới của ngôn ngữ lập trình để máy tính hiểu được.
* Một lớp thường có hai thành phần chính: các thuộc tính và các phương thức. Ngoài ra nó có thể chỉ có các phương thức, chỉ có thuộc tính hoặc có thêm các enum, các lớp, interface khác.
* Các đặc điểm của đối tượng sẽ được mô tả thông qua các thuộc tính của lớp.
* Các hành động của đối tượng sẽ được mô tả thông qua các phương thức.

**Cú pháp tổng quát**

* Sau đây là cú pháp tổng quát của một định nghĩa lớp trong Java:

**access non-access class name {**

***// properties***

***// constructors***

***// methods***

***// others***

**}**

* Trong đó:
  + Phần *access* là access modifier, tức khả năng nhìn thấy của một thành phần so với thế giới bên ngoài. Các access modifier có thể là public, protected, private. Mỗi lớp chỉ có 1 access modifier và nó chỉ xuất hiện 1 lần. Mỗi file chương trình Java chỉ có 1 và chỉ 1 public class. Ta sẽ lần lượt tìm hiểu kĩ hơn qua từng bài học.
  + *non-acess* là phần bổ trợ làm rõ nghĩa cho một lớp. Phần này có thể có hoặc không. *non-access* có thể là final, abstract, static, strictfp. Ta sẽ lần lượt tìm hiểu cụ thể trong các bài học. Bây giờ để cho đơn giản ta sẽ mặc định phần non-access là bỏ trống.
  + Tiếp theo là keyword *class*. Đây là keyword bắt buộc để khai báo và nhận biết lớp trong Java.
  + Phần *name* là tên của một lớp. Tên lớp là danh từ, cụm danh từ số ít mô tả đối tượng mà nó thể hiện. Tên lớp luôn bắt đầu bằng chữ cái hoa và nếu tên có nhiều từ, các từ viết liền nhau và viết hoa chữ cái đầu từ.
  + Trong cặp {} chứa các thành phần mô tả và bổ trợ nhằm làm rõ đối tượng cần mô hình hóa. Phần này có thể có nhiều loại thành phần cấu thành như trên hoặc cũng có thể chỉ có 1 thành phần bất kì trong số đó.
  + Phần properties là các thuộc tính, còn gọi là các trường(fields) là phần mô tả đặc điểm đặc trưng của đối tượng. Đây là thành phần dữ liệu của lớp. Giả định từ giờ đến khi học bài tính đóng gói dữ liệu ta sẽ coi rằng các thuộc tính là public.
  + Phần constructors là các phương thức khởi tạo dùng để tạo đối tượng với những thông tin ban đầu từ lớp.
  + Phần methods là các phương thức khác constructors. Dùng để thể hiện các hành động, nhiệm vụ, hành vi của đối tượng mà lớp mô tả.
  + Phần others có thể là các lớp, enum, interface. Để cho đơn giản thì phần này sẽ được đề cập chi tiết trong các bài học về sau.

**Ví dụ một số hình ảnh về lớp:**

*/\*\**

*\* lớp Student để mô tả thông tin sinh viên*

*\*/*

public class Student {

*// các thuộc tính:*

public String studentId;   *// mã sinh viên*

    public String fullName;    *// họ tên đầy đủ*

public float avgGrade;     *// điểm TB*

public String email;       *// địa chỉ email*

public String address;     *// địa chỉ cư trú*

*// constructors:*

*/\*\**

*\* phương thức khởi tạo mặc định*

*\*/*

public Student() {

    }

*/\*\**

*\* phương thức khởi tạo 1 tham số*

*\**

*\* @param name chứa tên sinh viên*

*\*/*

public Student(String name) {

        fullName = name;

    }

*/\*\**

*\* phương thức khởi tạo có hai tham số*

*\**

*\* @param name chứa tên sinh viên*

*\* @param id   chứa mã sinh viên*

*\*/*

public Student(String name, String id) {

        studentId = id;

        fullName = name;

    }

*// các phương thức khác:*

*/\*\**

*\* phương thức thể hiện hành động làm bài tập của SV*

*\* @param subject chứa tên môn học sinh viên cần làm bài tập*

*\*/*

public void doHomework(String subject) {

*// sinh viên đang làm bài tập...*

}

*/\*\**

*\* phương thức thể hiện sinh viên làm bài thi môn nào đó*

*\* @param subject chứa tên môn học cần làm bài thi*

*\*/*

public void doExam(String subject) {

*// sinh viên làm bài thi...*

}

}

*/\*\**

*\* lớp mô tả thông tin về con mèo*

*\*/*

public class Cat {

*// thuộc tính*

public String name;

    public int age;

    public float weight;

    public String color;

    public String favoriteFood;

*// constructor:*

public Cat(String name) {

        this.name = name;

    }

*// các phương thức:*

*/\*\**

*\* phương thức mô tả con mèo nói chuyện với nhau như nào*

*\*/*

public void speak() {

*// mèo nói chuyện...*

}

*/\*\**

*\* phương thức mô tả con mèo di chuyển như nào*

*\*/*

public void move() {

*// mèo đang bay qua nóc nhà...*

}

*/\*\**

*\* phương thức mô tả mèo ăn thức ăn nào đó ra sao*

*\**

*\* @param food món ăn nào đó như cá luộc chẳng hạn*

*\*/*

public void eat(String food) {

*// mèo ăn món food ngấu nghiến...*

}

}

**Cú pháp tạo và sử dụng đối tượng**

* Để sử tạo ra đối tượng ta dùng cú pháp: **type variable = new type(params);** với type là tên của lớp muốn sử dụng. params là danh sách tham số(có hoặc không).
* Ví dụ:

Cat tom = new Cat("Tom");

Student linh = new Student("Nguyễn Thùy Linh");

Student nhan = new Student("B25DCCN111", "Hoàng Trọng Nhân");

* Để truy cập đến một thành phần nào đó của lớp, ta sử dụng tên đối tượng và toán tử dấu chấm sau đó là tên thành phần: **name**. Trong đó name phải là thành phần public nào đó.
* Ví dụ:

Cat tom = new Cat("Tom");

Student linh = new Student("Nguyễn Thùy Linh");

Student nhan = new Student("B25DCCN111", "Hoàng Trọng Nhân");

*// hiển thị tên của mèo tom:*

System.*out*.println("Hello " + tom.name);

*// lấy mã sinh viên:*

System.*out*.println("Mã sinh viên: " + nhan.studentId);

*// cho sinh viên Linh đi thi môn Java:*

linh.doExam("Lập trình Java");

**Ví dụ minh họa**

Sau đây là ví dụ đầy đủ về lớp Student.java:

*/\*\**

*\* lớp Student để mô tả thông tin sinh viên*

*\*/*

public class Student {

*// các thuộc tính:*

public String studentId;   *// mã sinh viên*

public String fullName;    *// họ tên đầy đủ*

public float avgGrade;     *// điểm TB*

public String email;       *// địa chỉ email*

public String address;     *// địa chỉ cư trú*

*// constructors:*

*/\*\**

*\* phương thức khởi tạo mặc định*

*\*/*

public Student() {

    }

*/\*\**

*\* phương thức khởi tạo 1 tham số*

*\**

*\* @param name chứa tên sinh viên*

*\*/*

public Student(String name) {

        fullName = name;

    }

*/\*\**

*\* phương thức khởi tạo có hai tham số*

*\**

*\* @param name chứa tên sinh viên*

*\* @param id   chứa mã sinh viên*

*\*/*

public Student(String name, String id) {

        studentId = id;

        fullName = name;

    }

*// các phương thức khác:*

*/\*\**

*\* phương thức thể hiện hành động làm bài tập của SV*

*\**

*\* @param subject chứa tên môn học sinh viên cần làm bài tập*

*\*/*

public void doHomework(String subject) {

        System.*out*.println(fullName + " đang làm bài tập môn " + subject);

    }

*/\*\**

*\* phương thức thể hiện sinh viên làm bài thi môn nào đó*

*\**

*\* @param subject chứa tên môn học cần làm bài thi*

*\*/*

public void doExam(String subject) {

        System.*out*.println(fullName + " đang thi môn " + subject);

    }

*/\*\**

*\* phương thức để kiểm tra xem sinh viên có qua môn nào đó*

*\* với mức điểm TB cho trước không*

*\**

*\* @return true nếu SV qua môn và fale nếu ngược lại*

*\*/*

public boolean isPassed() {

        return avgGrade >= 4.0f;

    }

}

Lớp Test.java:

public class Test {

    public static void main(String[] args) {

        Student linh = new Student("Nguyễn Thùy Linh");

        Student nhan = new Student("B25DCCN111", "Hoàng Trọng Nhân");

*// lấy mã sinh viên:*

System.*out*.println("Mã sinh viên: " + nhan.studentId);

*// cho sinh viên Linh đi thi môn Java:*

linh.doExam("Lập trình Java");

*// gán điểm cho sinh viên:*

linh.avgGrade = 5.25f;

        nhan.avgGrade = 7.5f;

*// kiểm tra xem sinh viên trượt môn hay qua môn:*

System.*out*.println(linh.isPassed() ? "Linh qua môn" : "Linh trượt môn");

        System.*out*.println(nhan.isPassed() ? "Nhân qua môn" : "Nhân trượt môn");

    }

}

Kết quả:

Mã sinh viên: Hoàng Trọng Nhân

Nguyễn Thùy Linh đang thi môn Lập trình Java

Linh qua môn

Nhân qua môn