**Ý nghĩa và mục đích sử dụng**

* Tính đóng gói dữ liệu là 1 trong 4 tính chất của ngôn ngữ lập trình Java. Các tính chất còn lại là: tính kế thừa, tính đa hình và tính trừu tượng
* Tính đóng gói dữ liệu là kĩ thuật gói gọn các dữ liệu và các chức năng vận hành trên dữ liệu đó vào trong một đơn vị đơn nhất.
* Với tính đóng gói dữ liệu, dữ liệu của một lớp sẽ được bảo vệ khỏi thế giới bên ngoài. Chỉ những thành phần trong lớp đó mới nhìn thấy và sử dụng được.
* Nói cách khác, với tính đóng gói dữ liệu, dữ liệu của lớp nào chỉ lớp đó được phép trực tiếp thao tác. Các lớp khác muốn sử dụng hoặc thao tác với các dữ liệu này phải thông qua các phương thức public của lớp.
* Biểu hiện của tính đóng gói dữ liệu là:
  + Các thuộc tính(dữ liệu) sẽ có access modifier là private
  + Có các getter/setter public để thao tác với các thuộc tính private
* Ví dụ:

public class Student {

*// các thuộc tính:*

private String studentId;

    private String fullName;

    private float avgGrade;

    private String address;

    private String email;

*// constructors*

public Student() {

*//...*

}

*// getter*

    public String getStudentId() {

        return studentId;

    }

*// setter*

    public void setStudentId(String studentId) {

        if (!studentId.isEmpty()) { *// nếu mã sv không rỗng*

this.studentId = studentId; *// cập nhật mã*

}

    }

}

**Các getter/setter**

* Để đọc dữ liệu ra ta dùng getter
* Kiểu của getter luôn cùng kiểu với kiểu của thuộc tính cần đọc ra
* Tên của getter luôn có dạng: getX trong đó X là tên thuộc tính cần đọc
* Getter không nhận tham số
* Để thay đổi dữ liệu của đối tượng ta sử dụng setter
* Setter thường có kiểu void với định dạng setX với x là tên thuộc tính cần thay đổi giá trị.
* Phương thức setter luôn nhận vào một tham số cùng kiểu với kiểu của giá trị cần sửa đổi.
* Trong setter có thể thực hiện các hành động kiểm tra, so khớp trước khi gán giá trị để đảm bảo dữ liệu luôn hợp lệ.
* Một thuộc tính có thể có đầy đủ cả getter/setter hoặc cũng có thể chỉ có getter/setter
* Ví dụ:

public class Student {

*// các thuộc tính:*

private String studentId;

    private String fullName;

    private float avgGrade;

    private String address;

    private String email;

*// constructors*

public Student() {

        fullName = "";

        avgGrade = 0;

        studentId = "";

*//...*

}

    public String getStudentId() {

        return studentId;

    }

    public void setStudentId(String studentId) {

        if (!studentId.isEmpty()) { *// nếu mã sv không rỗng*

this.studentId = studentId; *// cập nhật mã*

}

    }

    public String getFullName() {

        return fullName;

    }

    public void setFullName(String fullName) {

        if (!fullName.isEmpty()) { *// nếu tên sv k rỗng*

this.fullName = fullName; *// cập nhật tên*

}

    }

    public Student(String id) {

        studentId = id;

    }

    public Student(String id, String name) {

        studentId = id;

        fullName = name;

    }

*// methods*

public void doHomework(String subject) {

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

    }

    public void doExam(String subject) {

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

    }

*//....*

}

**Các phương thức private**

* Tương tự như các thuộc tính private, phương thức private cũng chỉ có thể được sử dụng ở bên trong một lớp.
* Khi ta loại bỏ các private medthod sẽ không ảnh hưởng gì đến code bên ngoài. Vì đơn giản không có bất kì lời gọi nào tới phương thức xuất phát từ phía bên ngoài lớp hiện thời.
* Mục đích của các private methods là để thực hiện các xử lý nội bộ nào đó. Thường dùng để tạo các helper method nội bộ.
* Nếu một nhiệm vụ nào đó có nhiều vấn đề nhỏ hơn cần xử lý thì ta có thể chia các vấn đề đó thành các private method để xử lý.
* Khi sinh code trong công cụ IntelliJ ta thấy rằng các phương thức sinh ra mặc định là private. Nếu muốn phương thức này được sử dụng bởi các thành phần bên ngoài lớp, thì ta chỉ cần đổi private thành một trong các access modifier khác tùy mục đích.
* Ví dụ:

public void setFullName(String fullName) {

    if (isValidName(fullName)) { *// nếu tên sv k rỗng*

this.fullName = fullName; *// cập nhật tên*

} else {

        System.*out*.println("Tên không hợp lệ");

    }

}

private boolean isValidName(String fullName) {

*// nếu tên rỗng, không hợp lệ*

if(fullName.length() == 0) {

        return false;

    }

*// nếu tên chứa kí tự số hoặc kí tự đặc biệt*

*// thì không hợp lệ*

if(fullName.matches(".\*[^a-zA-Z\\s]+.\*")) {

        return false;

    }

    return true; *// mặc định tên hợp lệ*

}

**Ví dụ minh họa**

Sau đây là code mẫu đầy đủ của lớp Student.java:

*/\*\**

*\* lớp mô tả thông tin sinh viên*

*\*/*

public class Student {

*// các thuộc tính:*

private String studentId;

    private String fullName;

    private float avgGrade;

    private String address;

    private String email;

*// constructors*

public Student() {

        fullName = "";

        avgGrade = 0;

        studentId = "";

*//...*

}

    public String getStudentId() {

        return studentId;

    }

    public void setStudentId(String studentId) {

        if (!studentId.isEmpty()) { *// nếu mã sv không rỗng*

this.studentId = studentId; *// cập nhật mã*

}

    }

    public String getFullName() {

        return fullName;

    }

    public void setFullName(String fullName) {

        if (isValidName(fullName)) { *// nếu tên sv k rỗng*

this.fullName = fullName; *// cập nhật tên*

} else {

            System.*out*.println("Tên không hợp lệ");

        }

    }

    private boolean isValidName(String fullName) {

*// nếu tên rỗng, không hợp lệ*

if(fullName.length() == 0) {

            return false;

        }

*// nếu tên chứa kí tự số hoặc kí tự đặc biệt*

*// thì không hợp lệ*

if(fullName.matches(".\*[^a-zA-Z\\s]+.\*")) {

            return false;

        }

        return true; *// mặc định tên hợp lệ*

}

    public Student(String id) {

        studentId = id;

    }

    public Student(String id, String name) {

        studentId = id;

        fullName = name;

    }

*// methods*

public void doHomework(String subject) {

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

    }

    public void doExam(String subject) {

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

    }

*//....*

}

Code của lớp dùng để test: Bai29.java

public class Bai28 {

    public static void main(String[] args) {

Student s = new Student();

        s.setFullName("Tran van hung6"); *// Thay bởi Tony Stark*

        System.*out*.println(s.getFullName());

    }

}

Kết quả:

Tên không hợp lệ

------------------------

Tony Stark