Chương 06:

Lập trình hướng đối tượng nâng cao

1. *Abstract class*
2. *Interface*
3. *So sánh Abstract và Interface*
4. *Sử dụng class Object*
5. *Giảm thiểu sự phụ thuộc giữa các class*
6. *Inner Class trong Java*
7. *Xây dựng ứng BookStore Version 5*

Abstract class

- *Abstract class (lớp trừu tượng) là class chứa bộ khung cơ bản, với các thuộc tính, các phương thức và các phương thức trừu tượng (abstract method phương thức chưa hoàn thiện: chỉ có tên, kiểu trả về, tham số)*
- *Các lớp extends từ lớp abstract sẽ có nhiệm vụ hoàn thiện các abstract method*
- *Abstract class thường được xây dựng bởi kỹ sư thiết kế hệ thống có kinh nghiệm*

Interface

- *Interface không phải là class, đây là một mẫu giao diện quy định một số phương thức bắt buộc cho một class nào đó*
- *Interface không cho phép định nghĩa rõ cách hoạt động của phương thức, chỉ dừng lại ở khai báo phương thức, việc định nghĩa các phương thức này sẽ được thực hiện ở các class con*

Sử dụng default và static trong Interface

- *Tạo mới một phương thức trong Interface → định nghĩa cách hoạt động của phương thức này ở các class implements từ Interface đó.*
- *Sử dụng từ khóa default hoặc static để khi tạo mới một phương thức thì không cần phải định nghĩa ở tất cả các class implements từ Interface.*
- *Sử dụng default method thì cần khởi tạo đối tượng <> Sử dụng static method thì không cần khởi tạo đối tượng*

Abstract Class vs Interface

Những điểm giống nhau

- *Chứa các abstract methods*
- *Không khởi tạo trực tiếp đối tượng mà phải thông qua các class con*

Abstract Class vs Interface

Abstract class

+ Là một lớp

*+ Mỗi method **không bắt buộc** phải được định nghĩa ở lớp con*

+ Mỗi lớp con chỉ kế thừa từ một lớp abstract, một lớp abstract có nhiều lớp con

+ Mỗi phương thức abstract có thể ở trạng thái protected hoặc public

Interface

+ Không phải lớp

*+ Mỗi method **bắt buộc** phải được định nghĩa ở lớp con*

+ Mỗi lớp con kế thừa từ một hoặc nhiều Interface, một Interface có nhiều lớp con

+ Mỗi phương thức abstract chỉ ở trạng thái public

Sử dụng class Object

- *Tất cả các class trong Java (có sẵn hay do lập trình viên định nghĩa) đều kế thừa từ class Object*
- *Sử dụng phương thức* **toString()** *để in ra thông tin của một đối tượng nào đó*
- *Sử dụng phương thức* ***equals()*** *để so sánh các đối tượng với nhau*

Tình huống thực tế "Đi công tác"

Tình huống thực tế “Đi công tác”

Phòng Quản Lý

Giảm thiểu sự phụ thuộc giữa các Class

- *Giảm thiểu sự phụ thuộc giữa các Class tạo ra sự linh hoạt cao và tính dễ bảo trì cho ứng dụng*
- *Nếu giảm thiểu tốt sự phụ thuộc sẽ giúp tăng khả năng dung lại và khả năng dễ đọc của mã nguồn, tạo cho ứng dụng tính mềm dẻo và khả năng nâng cấp bảo trì trở nên đơn giản và nhanh chóng hơn*

Inner Class trong JAVA

Inner Class là một class được khai báo và tồn tại bên trong một class khác

- *Tạo nhóm quan hệ giữa các class với nhau*
- *Các class lồng nhau tăng tính dễ đọc và bảo trì cho ứng dụng*
- *Mã nguồn gắn gọn và đơn giản hơn*

Inner Class trong JAVA

Type	Description
Member Inner Class	Class nằm trong class và ngoài các method
Anonymous Inner Class	Class ẩn danh, được tạo ra để hiện thực các interface hoặc các class mở rộng khác
Local Inner Class	Class nằm trong method
Static Nested Class	Static class được tạo ra bởi một class
Nested Interface	Interface được tạo ra bởi một class hoặc interface