**Access Modifier**

- Quy định mức truy cập lớp và các thành phần của lớp

- public: truy xuất ở bất kỳ đâu

- private: truy xuất trong cùng lớp

- protected: truy xuất trong cùng lớp và lớp con

- default: khi không có các quy định trên -> default -> truy xuất trong cùng package(gói)

Mức độ truy xuất: public>protected>no modifier(default)>private

Data field encapslution (bao gói trường dữ liệu) -> hạn chế truy cập trực tiếp bằng từ khóa private

- khai báo getter:

+ Thông thường:

**Public kiểu dữ liệu trả về get\_tên thuộc tính();**

+ Kiểu boolean:

**Public boolean is\_tên thuộc tính();**

- khai báo setter:

**Public void set\_tên thuộc tính(kiểu dữ liệu + giá trị thuộc tính)**

\* Lưu ý: nếu tên tham số và thuộc tính trùng nhau thì dùng từ khóa this để phân biệt.

VD:

class Person {

public String name; //thuộc tính

public Person(String name) { //tham số

this.name = name;

}

}

Package và Static

**Package**

- Phân loại lớp và interface thành nhóm liên quan -> đơn vị để dễ quản lý

- các gói trong package:

+ java.io: nhập xuất dữ liệu

+ java.net: thao tác qua mạng lưới

+ java.security: thao tác liên quan đến bảo mật

+ java.util: cung cấp lớp và các phương thức hỗ trợ

- tính chất:

+ có thể có gói con (subpackage) trong package

+ không thể có 2 lớp cùng tên trong cùng 1 gói

+ package(lớp) thì dùng tên gói để truy cập vào lớp

+ bắt đầu bằng từ khóa java hoặc javax

- khai báo:

**Package tên gói**

+ Tên gói phải trùng tên thư mục chứa mã nguồn

+ subpackage phải lần lượt trùng tên thư mục tương ứng

VD:

package com.codegym.ui; //ui ở trong codegym,codegym ở trong com

Import

- Dùng từ khóa import để sử dụng các lớp trong các package khác

VD:

package model;

public class Customer {

}

package controller;

import model.Customer;

public class CustomerController {

public void index(){

Customer customer = new Customer();

}

}

**Static**

- Dùng để khai báo thuộc tính và phương thức của lớp

- trực thuộc lớp

- Có thể truy xuất thành phần của static bằng cách sử dụng lớp /đối tượng

- Không có đối tượng vẫn sử dụng được các thành phần của static

- khai báo static property:

**Quy định truy xuất(modifier) kiểu dữ liệu tên biến**

VD:  
Khai báo biến static:

class Application{

public static String language = "english";

}

Truy xuất biếnstatic:

System.out.println("Current language: " + **Application.language**)

- khai báo static method:

**Quy định truy xuất static kiểu dữ liệu tên phương thức()**

VD:

Khai báo phương thức static

class Application{

public static String getVersion(){

return "1.0";

}

}

Gọi phương thức static

System.out.println("Current version: " + Application.getVersion());

- ràng buộc của statcic:

+ chỉ có thể gọi phương thức static khác

+ chỉ có thể truy xuất các biến static khác

+ không sử dụng this/super vì nó được dùng cho đối tượng

+ có thể khởi tạo static thông qua khối khởi tạo

VD:

class Application{

public static String language;

**static {**

**if(System.getProperty("lang").equals("en")){**

**language = "english";**

**} else {**

**language = "spanish";**

**}**

**}**

}

Bảng sau thể hiện phạm vi truy cập của các access modifier:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phạm vi** | **Public** | **Protected** | **Default** | **Private** |
| Từ trong cùng một lớp (class) | YES | YES | YES | YES |
| Từ bất kì lớp (class) nào trong cùng một gói (package) | YES | YES | YES | NO |
| Từ một lớp con (subclass) trong cùng một gói (package) | YES | YES | YES | NO |
| Từ một lớp con (subclass) bên ngoài gói (package) | YES | YES, thông qua kế thừa | NO | NO |
| Từ bất kì một lớp(class) bên ngoài gói (package) | YES | NO | NO | NO |

Nested Class

Inner class

Inner class: là một class được khai báo bên trong một class khác. Inner class có thể truy cập vào tất cả các biến và phương thức của lớp bao bọc nó. Inner class có thể được sử dụng để giải quyết các vấn đề liên quan đến việc phân tách một class lớn thành các thành phần nhỏ hơn có thể tái sử dụng được.

VD:

public class OuterClass {

private int x = 10;

class InnerClass {

public void display() {

System.out.println("Value of x is " + x);

}

}

public static void main(String[] args) {

OuterClass outerObject = new OuterClass();

OuterClass.InnerClass innerObject = outerObject.new InnerClass();

innerObject.display();

}

}

Static nested class

Static nested class: là một class được khai báo bên trong một class khác với từ khóa static. Static nested class không thể truy cập các biến không phải là static của lớp bao bọc nó. Static nested class thường được sử dụng để nhóm các lớp có liên quan với nhau.

VD:

public class OuterClass {

static class StaticNestedClass {

public void display() {

System.out.println("Static Nested Class");

}

}

public static void main(String[] args) {

OuterClass.StaticNestedClass nestedObject = new OuterClass.StaticNestedClass();

nestedObject.display();

}

}