**Singleton**

**I. Định nghĩa**

Singleton là một pattern có 2 tiêu chí

1. Giới hạn chỉ có duy nhất 1 instance của class đó tồn tại ở bất kỳ thời điểm nào class đó được request
2. Cho phép quyền truy cập global

**II. Lý do sử dụng**

Chương trình cần đối tượng chỉ có một thể hiện duy nhất. Cùng với đó là khả năng có thể truy cập đối tượng mọi lúc, mọi nơi và lazy-init cũng rất cần thiết.

Khi bạn cần đối tượng thực hiện chức năng cụ thể, như ghi log cho chương trình, nó nên được viết thành Singleton. Bất kể tại phần nào trong chương trình, chỉ có một thể hiện, chỉ có một cách truy cập đến Logger.

Để biến một class thành Singleton, cần đảm bảo rằng:

* Định nghĩa một attribute là private static và đó là thể hiện duy nhất của class này
* Định nghĩa public static getInstance() dùng để khởi tạo đối tượng (hàm accessor)
* Thực hiện lazy-init trong hàm accessor (chỉ khi gọi mới khởi tạo thể hiện)
* Constructor (hoặc các constructor) là private hay protected, vì bạn không muốn client tạo nhiều thể hiện
* Client chỉ có thể gọi hàm accessor khi muốn có thể hiện của class

**III. Ví dụ**

public class Singleton {

// 1. Set instance thành static

private static Singleton singletonInstance = new Singleton();

// 2. Constructor set về private

private Singleton() {

}

// 3. Getter set thành static method

public static Singleton getSingletonInstance() {

return singletonInstance;

}

}

Như vậy cách thức implement tổng quát là

1. Tạo một private static instance của singleton class đó, đây chính là instance duy nhất của class này
2. Tạo private constructor, không cho phép tạo mới instance từ bên ngoài singleton class
3. Tạo một public static method giúp những class bên ngoài lấy ra instance duy nhất của singleton class này