## http://viettuts.vn/java-thread

## Đa nhiệm (multitasking)

là một quá trình thực hiện nhiều nhiệm vụ cùng một lúc. Chúng ta sử dụng đa nhiệm để tận dụng tính năng của CPU. Đa nhiệm có thể đạt được bằng hai cách:

* Đa nhiệm dựa trên tiến trình (Process) – Đa tiến trình (Multiprocessing)
* Đa nhiệm dựa trên luồng (Thread) – Đa luồng (MultiThreading)

### **Đa tiến trình (Multiprocessing)**

* Mỗi tiến trình có địa chỉ riêng trong bộ nhớ, tức là mỗi tiến trình phân bổ vùng nhớ riêng biệt.
* Sự giao tiếp giữa các tiến trình có tri phí cao.
* Chuyển đổi từ tiến trình này sang tiến trình khác đòi hỏi thời gian để đăng ký việc lưu và tải, các bản đồ bộ nhớ, các danh sách cập nhật, vv.

**Đa luồng (multithreading) trong java** là một tiến trình thực hiện nhiều luồng đồng thời. sử dụng đa luồng nhiều hơn đa tiến trình bởi vì các luồng chia sẻ một vùng bộ nhớ chung. Chúng không phân bổ vùng bộ nhớ riêng biệt để tiết kiệm bộ nhớ, và chuyển đổi ngữ cảnh giữa các luồng mất ít thời gian hơn tiến trình.

**Ưu điểm của đa luồng trong java**

1) Nó không chặn người sử dụng vì các luồng là độc lập và bạn có thể thực hiện nhiều công việc cùng một lúc.

2) Bạn có thể thực hiện nhiều hoạt động với nhau để tiết kiệm thời gian, chi phí.

3) Luồng là độc lập vì vậy nó không ảnh hưởng đến luồng khác nếu ngoại lệ xảy ra trong một luồng duy nhất.

public class mythread1 implement Runnable{

@Override

public void run(){ -> ghi đè phương thức có sẵn}

public class mythread2 extends Thread{}

@Override

public void run(){ -> ghi đè phương thức có sẵn}

mythread1 my1 = new mythread1();

Thread m1 = new Thread(my1); // vì implement interface nên phải tạo Thread mới

mythread2 my2 = new mythread2();

m1.start();

my2.start();

t1.join(1000); //đợi thread t1 chạy xong hết rồi mới đến thread khác.

t1.setPriority(n); //ko có ý nghĩa,  nó phụ thuộc vào thông số kỹ thuật của JVM

t1.setDaemon(true); //thread chạy ngầm, dọn rác

**synchronized**: 2 thread đồng bộ, cùng truy xuất 1 hàm, dùng syn để khóa lại, nếu thread1 đã chiếm giữ rồi thì thread2 phải đợi

Thread.currentThread().getName()

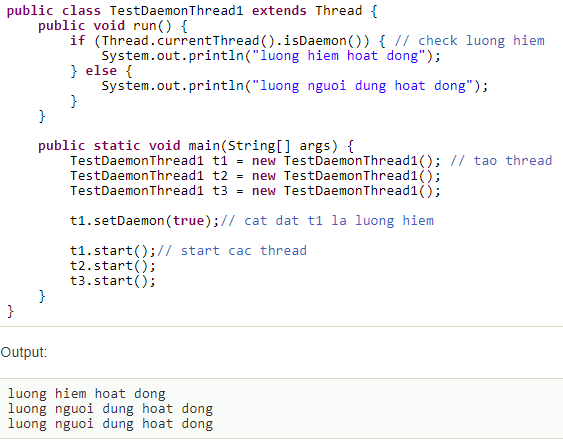
Thread.currentThread().getPriority()

## Có thể start một thread hai lần không?

Không. Sau khi start một thread, nó không bao giờ có thể được start lại. Nếu bạn làm như vậy, một ngoại lệ IllegalThreadStateException sẽ xảy ra.

**Luồng hiểm trong java (thread daemon)** **là một luồng cung cấp dịch vụ cho luồng người dùng**. Thời gian sống của nó phụ thuộc vào các luồng người dùng. Tức là khi tất cả các luồng người dùng chết, JVM sẽ tự động chấm dứt luồng này.

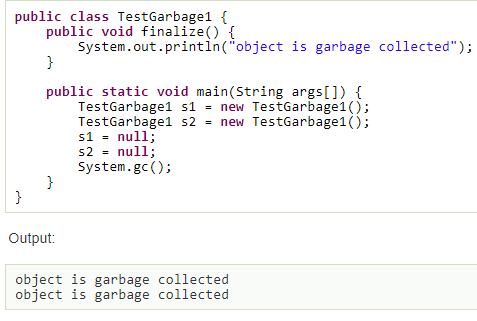
Có nhiều luồng hiểm java chạy tự động, ví dụ: Gc, finalizer, v.v.



## Phương thức finalize()

Phương thức finalize() được gọi mỗi lần trước khi đối tượng được thu gom rác. Phương thức này có thể được sử dụng để thực hiện xử lý dọn dẹp. Phương thức này được định nghĩa trong lớp Object như sau:

Ghi chú: Garbage Collection được thực hiện bởi một luồng hiểm gọi là Garbage Collector (GC). Luồng này gọi phương thức finalize() trước khi đối tượng thu gom rác.



**Đồng bộ trong java** (Synchronization in java) là khả năng kiểm soát truy cập của nhiều luồng đến bất kỳ nguồn tài nguyên chia sẻ (shared resource).

 Java Synchronization là lựa chọn tốt khi chúng ta muốn cho phép chỉ một luồng được truy cập vào tài nguyên chia sẻ đó.

## Tại sao sử dụng Synchronization

Việc đồng bộ hóa chủ yếu được sử dụng để:

1. Để tránh sự can thiệp của luồng khác.
2. Để đảm bảo vấn đề nhất quán của chương trình.