**Advanced Java: Multi-threading Part 2 -- Basic Thread Synchronization**

https://www.youtube.com/watch?v=\_aNO6x8HXZ0

**import** java.util.Scanner;

**class** Processor **extends** Thread{

**private** **volatile** **boolean** running=**true**; //Tutorial concerns **volatile**

**public** **void** run(){

**while**(running){

System.*out*.println("Hello");

**try** {

Thread.*sleep*(100);

} **catch** (InterruptedException e) {

e.printStackTrace();

}

}

}

**public** **void** shutdown(){

running=**false**;

}

}

**class** apples {

**public** **static** **void** main(String args[]){

Processor proc1 = **new** Processor();

proc1.start();

System.*out*.println("Press return to stop...");

Scanner scanner = **new** Scanner(System.*in*);

scanner.nextLine();

proc1.shutdown();

}

}

**Result:**

Press return to stop...

Hello

Hello

Hello

Hello

Hello

🡨 Pressed enter at this point

Hello

**Explanation:** Initially, programmer made a simple program that displays hello every 0.1 sec. After that, the programmer proposed a way to end the loop gracefully. How? By changing the value of running into **false**. But here’s the catch:

**There will be 2 threads.One in the main class, other in the original thread. The original thread doesn’t expect its code to change due to other threads. So the main thread will stop, while the original thread will still be looping since it doesn’t know its value running is not true anymore.**

That’s where the keyword “**volatile**” comes in. Due to the keyword, the thread can be affected by other threads.