Thread 강의

- ***Window)*** 프로세스 내부에 여러 스레드가 존재

- ***UNIX(MAC), LINUX)*** 프로세스와 스레드가 서로 별개, 따로 관리

  (but, 프로세스가 대빵, 스레드는 프로세스에 소속된 대원 정도로 관리)

- 프로세스 : CPU의 추상화 -> CPU가 기계어를 구동시켜야 프로그램 코드가 돌아감

- 추상화? 하나씩 시분할 방식으로 동작되는 것을 -> 프로그램이 동시에 실행하는 것처럼

  느끼게 만듬(프로세스들이 우선순위를 가지고 하나씩 동작함)

- main()을 sleep()하고자 한다면 반드시!! Thread 동작이 완료된 이후에 sleep() 시키기

QT Thread 동작을 완료한다는 것이 정확히 언제인지 모르겠습니다(ThirdThreadTest 클래스)

FirstThread 클래스(1)

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

- main()이 프로세스(스레드들의 대장)가 된다.

```
Thread t1 = new Thread(new Worker("대머리독수리"));
```

- new Thread(Worker()) -> 프로세스에 소속된 스레드 하나 생성

```
t1.start();
```

- 스케줄러에 1개의 스레드 등록(스레드를 생성하면 등록 후 run()이 구동됨으로써 실행)

```
class Worker implements Runnable {
```

new Thread를 하여 implement한 객체를 생성하면, 무조건 Thread가 CPU를 획득하기

FirstThread 클래스(2)

```
Random generator = new Random();
```

- generator라는 난수 생성 객체 선언

```
private final static Random generator = new Random();
```

- 선언 후 JVM 메모리에 올라가면 같은 값을 클래스 내부 전체 필드, 메소드에서 공유 /

- 필드 or 메소드에서 호출 시마다 새로운 값이 할당 xxx 같은 값으로 계속.

```
sleepTime = generator.nextInt(3000) + 500;
```

```
Thread.sleep(sleepTime);
```

- '스케줄링할 때 0.5ms ~ 3.5ms동안 현재 프로세스는 잠시 동작을 멈춘다"라는 뜻을 내포

(시분할 스케줄링에서 아직 일처리할 것들이 남았지만 시간이 만료되어 제어권이 다른 스레드에게 양도되었을 때, 이 양도한 스레드는 wait queue 대기열에서 차례가 올 때까지 대기.

but, sleep 시 대기열에도 존재 X, 잠시 동작을 멈춤 --> 해당 시간 끝난 후 대기열로 복귀)

FirstThread 클래스(3)

```
Thread t1 = new Thread(new Worker("대머리독수리"));
```

```
Thread t2 = new Thread(new Worker("대갈장군"));
```

```
Thread t3 = new Thread(new Worker("뿌뿌뿡"));
```

- 해당 스레드 생성 후 t1.start(), t2.start(), t3.start()를 통해 스레드가 실행(run()이 구동)

되면 thread가 CPU를 획득하기 위해 경쟁 → t1, t2, t3순으로 실행이 아닌 우선순위

높은 것 먼저, 시분할방식으로 실행

→실행 결과 : 대갈장군 -> 뿌뿌뿡 -> 대머리독수리 / 대머리독수리 -> 대갈장군 -> 뿌뿌뿡

SecondThread 클래스(1)

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

- 클래스 실행시 main()이 프로세스가 된다.

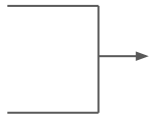
```
Thread t1 = new Thread(new SecondThreadTest("t1"));
```

```
Thread t2 = new Thread(new SecondThreadTest("t2"));
```

- 스레드로 만들 내용을 두 개(t1, t2)로 준비 -> 우선순위 높은 것이 먼저 실행

```
t1.start();
```

```
t2.start();
```



스케줄러에 2개의 스레드 등록

```
System.out.println("main() 프로세스 실행 중");
```

- 현재 아직 대장(main) 프로세스에게 할당한 시간 남아있어 여기까지 실행 후, 할 일 끝내고 대기(스레드들에게 양도)

SecondThread 클래스(2)

```
public class SecondThreadTest implements  Runnable {
```

- 스레드 실행되도록 run() 메소드 구동됨

```
for(int i = 1; i < 3; i++) {
```

```
   System.out.println(name + ": " + random.nextInt(100));
```

```
   try {
```

```
         Thread.sleep(500);
```

- random.nextInt(100) : 0~99의 랜덤 정수형 출력

- sleep()은 0.5초 → 0.5초 후 출력

- 여러 프로세스가 경쟁을 하다가 t1, t2 중 우선순위 높은 것이 먼저 출력

<결과값 ex>
t2 : 78
t1 : 80
t1 : 93
t2 : 24

ThirdThread 클래스(1)

```
t2.setDaemon(true);
```

- 데몬 프로세스 : 주 스레드 종료시 데몬 스레드는 강제 종료

t2.start() 전에 먼저 호출되어야 함

```
public static void main(String[] args) {
    t1.start();
    try {
        Thread.sleep(800);
    } catch (InterruptedException e) {
    }
    t2.start();
    System.out.println("main() 실행 중");
```

main() 메소드 내에서 sleep(800) 사용시

→ 0.8초 후(t1의 run()이 8번 실행된 후)

t2.start() 메소드 실행

+

main()이 실행 중 출력