제 42강

스레드

교재: p238~242

목차

1. 스레드의 정의

스레드란?

: 하나의 프로세스 안 에서 두 가지 이상의 일을 하도록 하는 것

<용어 정리>

- 프로세스(Process) : 실행 중인 프로그램
- 스레드(Thread) : 프로세스에서 작업을 수행하는 것
- 멀티 스레드 프로세스(Multi-Thread Process)
  : 두 가지 이상의 작업을 하는 프로세스

1. 스레드의 정의

[프로세스가 실행되는 방식]

1. 시간 분할 방식

: 모든 프로세스에게 동일한 시간을 할당하고 골고루 실행되는 방식

2. 선점형 방식

: 각각의 프로세스에게 우선 순위를 부여하고 우선순위가 높은 순으로 실행되는 방식

2. 스레드의 생성

[JVM이 스레드 처리시 하는 일] => 스레드 스케줄링

- 스레드가 몇 개 존재하는지
- 스레드가 실행되는 프로그램 코드의 메모리 위치가 어디인지
- 현재 스레드의 상태는 무엇인지
- 스레드의 우선순위는 몇 인지

[개발자가 스레드 처리시 하는 일]

- 자바 스레드로 작동할 작업이 무엇인지 코드로 작성
- 스레드 코드가 실행할 수 있도록 JVM 한테 요청

2. 스레드의 생성

1. 스레드 작업 코드 작성 방법

1) Thread 클래스 상속

2) Runnable 인터페이스 구현

---

1) Thread 클래스 상속

```
class T1 extends Thread{
        public void run( ){   run( )메서드 오버라이딩
                                    //작업할 내용
                        }
}
```

2. 스레드의 생성

1. 스레드 작업 코드 작성 방법

1) Thread 클래스 상속
2) Runnable 인터페이스 구현

---

2) Runnable 인터페이스 구현

```
class T2 implements Runnable{
        public void run( ){   run( )메서드 오버라이딩
                                    //작업할 내용
                        }
}
```

2. 스레드의 생성

2. 스레드 코드 실행하도록 JVM 요청

1) Thread 클래스를 상속받은 클래스
2) Runnable 인터페이스 구현한 클래스

---

1) Thread 클래스를 상속받은 클래스

```
Th1 t1 = new Th1( ); //인스턴스 생성
t1. start( );
```

2. 스레드의 생성

2. 스레드 코드 실행하도록 JVM 요청

1) Thread 클래스를 상속받은 클래스
2) Runnable 인터페이스 구현한 클래스

---

2) Runnable 인터페이스 구현한 클래스

```
Th2 t2 = new Th2( ); //인스턴스 생성
Thread t = new Thread(t2);

t.start( );
```

Thread 클래스의 매개변수 생성자에
인자값으로 전달!

2. 스레드의 생성

<실습> Exam-93.java

Thread 클래스 상속과 Runnable 인터페이스 구현한 스레드를 각각 실행하는 실습

```
class MyThread extends Thread{
    public void run() {
        for(int i=0;i<10;i++) {
            System.out.println("Thread 진행 중 "+i);
        }
    }
}
class MyThread2 implements Runnable{
    @Override
    public void run() {
        for(int i=0;i<10;i++) {
            System.out.println("Runnable 진행 중 "+i);
        }
    }

}
```

```
public class Thread1 {
    public static void main(String[] args) {
        MyThread t1= new MyThread();
        MyThread2 t2= new MyThread2();

        t1.start();

        Thread t= new Thread(t2);
        t.start();

        for(int i=0;i<10;i++) {
            System.out.println("main 진행 중 "+i);
        }
    }
}
```

main도 스레드

실행결과, 스레드를 호출한 순서에 상관 없이
메인 함수, 러너블, 스레드가 골고루 실행되고 있다.

<실행결과>

```
Thread 진행 중 0
Thread 진행 중 1
Thread 진행 중 2
main 진행 중 0
Runnable 진행 중 0
Thread 진행 중 3
Runnable 진행 중 1
Thread 진행 중 4
Thread 진행 중 5
Runnable 진행 중 2
Thread 진행 중 6
Thread 진행 중 7
Thread 진행 중 8
Thread 진행 중 9
main 진행 중 1
Runnable 진행 중 3
Runnable 진행 중 4
Runnable 진행 중 5
Runnable 진행 중 6
Runnable 진행 중 7
Runnable 진행 중 8
Runnable 진행 중 9
main 진행 중 2
main 진행 중 3
main 진행 중 4
main 진행 중 5
main 진행 중 6
main 진행 중 7
main 진행 중 8
main 진행 중 9
```

2. 스레드의 생성

<실습> MainThread.java

Thread 이름, 상태, 우선순위 확인하는 메서드 실습

```
public class MainThread {
    public static void main(String[] args) {
        Thread3 t3= new Thread3();
        t3.start();

        System.out.println("Thread Name:"+Thread.currentThread().getName());
        System.out.println("Thread state:"+Thread.currentThread().getState());
        System.out.println("Thread priority:"+Thread.currentThread().getPriority());

    }
}

class Thread3 extends Thread{
    public void run() {
        this.setName("Thread3");
        System.out.println("Thread Name:"+Thread.currentThread().getName());
        System.out.println("Thread state:"+Thread.currentThread().getState());
        System.out.println("Thread priority:"+Thread.currentThread().getPriority());
    }
}
```

<실행결과>

```
Thread Name:Thread3
Thread state:RUNNABLE
Thread priority:5
Thread Name:main
Thread state:RUNNABLE
Thread priority:5
```