Thread Ⅱ - Characteristic (Scheduling)

쓰레드의 스케쥴링과 우선순위(Priority) 컨트롤

- 우선순위가 높은 쓰레드의 실행을 우선시한다.
- 우선순위는 숫자로 표현한다 (7>1, 우선순위 7인 쓰레드가 더 높다.)
- 우선순위가 동일할 때에는 CPU의 할당시간을 나눈다.

---

i. 우선순위가 동일한 경우

CPU가 Thread를 번갈아가며 실행

```
public class MessageSendingThread extends Thread {
	String message;

	public MessageSendingThread(String str) {
		message = str;
	}

	@Override
	public void run(){
		for(int i=0; i<1000000; i++){
			System.out.println(message+"("+getPriority()+")");
			// getPriority() : 쓰레드의 우선순위를 int형으로 반환
		}
	}
}

public class PriorityTestOne {
	public static void main(String[] args) {
		MessageSendingThread thread1 = new MessageSendingThread("first");
		MessageSendingThread thread2 = new MessageSendingThread("second");
		MessageSendingThread thread3 = new MessageSendingThread("third");

		thread1.start();
		thread2.start();
		thread3.start();
	}
}
```

```
-----------------------------------------------------------------------------------------
Command Prompt

first(5)
first(5)
first(5)
first(5)
second(5)
third(5)
second(5)
second(5)
third(5)
third(5)
first(5)
third(5)
. . . . .
-----------------------------------------------------------------------------------------
```

---

ii. 우선순위가 다를 경우

우선순위가 가장 높은 Thread가 가장 먼저 실행되고 종료되어야 그 다음 우선순위의 Thread가 실행

```
public class MessageSendingThread extends Thread {
     String message;

     public MessageSendingThread(String str, int prio) {
          message = str;
          setPriority(prio);
          // setPriority() : 쓰레드의 우선순위를 지정하는 메소드
     }

     @Override
     public void run(){
          for(int i=0; i<5; i++){
               System.out.println(message+"("+getPriority()+")");
          }
     }
}

public class PriorityTestTwo {
     public static void main(String[] args) {
          MessageSendingThread thread1
               = new MessageSendingThread("first", Thread.MAX_PRIORITY);
          MessageSendingThread thread2
               = new MessageSendingThread("second", Thread.NORM_PRIORITY);
          MessageSendingThread thread3
               = new MessageSendingThread("third", Thread.MIN_PRIORITY);

          Thread.class
```

```
// Field descriptor #41 I
public static final int MIN_PRIORITY = 1;

// Field descriptor #41 I
public static final int NORM_PRIORITY = 5;

// Field descriptor #41 I
public static final int MAX_PRIORITY = 10;
```

```
          thread1.start();
          thread2.start();
          thread3.start();
     }
}
```

Command Prompt

```
first(10)
first(10)
first(10)
. . . . .
second(5)
second(5)
second(5)
. . . . .
third(1)
third(1)
```

```
third(1)
. . . . .
--------------------------------------------------------------------------------------------
```

---

i. 낮은 우선순위의 쓰레드 실행

```
public class MessageSendingThread extends Thread {
     String message;

     public MessageSendingThread(String str, int prio) {
          message = str;
          setPriority(prio);
     }

     @Override
     public void run(){
          for(int i=0; i<1000000; i++){
               System.out.println(message+"("+getPriority()+")");

               try{

                    Thread.sleep(1);
                    // 악간의 딜레이 시간으로 해당 Thread는 실행하지 않는 상태가 된다.
                       그 순간 다른 Thread에게 CPU를 양보한다!

               }catch (InterruptedException e) {
                    e.printStackTrace();
               }
          }
     }
}

--------------------------------------------------------------------------------------------
Command Prompt

first(10)
third(1)
second(5)
first(10)
third(1)
second(5)
first(10)
third(1)
second(5)
first(10)
. . . . .
--------------------------------------------------------------------------------------------
```
