업무에 바로 쓰는 JAVA 프로그래밍 입문

13. 라이브러리 활용하기

스레드

1 개요

› 스레드는 경량 프로세스로서 동시에 여러 개의 명령문들을 실행할 수 있음

2 스레드 생성

› Thread 클래스
- 스레드 형태로 실행할 객체를 구현할 때 상속받는 객체
- 스레드로서 동작할 실행문은 run( ) 메소드에 구현함
- run( ) 메소드 실행 : 객체 생성 후 start( )메소드를 호출

› Runnable 인터페이스
- 스레드 형태로 실행할 객체를 구현할 때 상속받는 객체
- 스레드로서 동작할 실행문은 run( ) 메소드에 구현함
- run( ) 메소드를 실행하기 위해 Thread 클래스를 생성해야 함
- Thread 클래스 생성시 인자값으로 Runable 상속하는 객체를 전달함
- run( ) 메소드 실행 : 생성된 Thread 클래스의 start( )메소드를 호출

› Thread 설정
- 스레드를 제어하기 위해 스레드에 setName( )으로 이름을 지정할 수 있음
- 동시에 실행되는 여러 스레드는 setPriority( )로 실행되는 우선 순위를 지정할 수 있음

3 동기화

› 동기화는 하나의 자원을 여러 스레드가 동시에 사용할 때 자원의 안전성을 확보하기 위해 처리하는 기능임

› 특정객체
- synchronized(객체 이름)로 특정 객체에 대하여 동기화를 지정할 수 있음

› 메소드
- 메서드 선언부에 synchronized를 선언하여 메소드 단위로 동기화를 지정할 수 있음

› 스레드 상태

Runnable 상태	Running 상태	Terminated 상태
실행하기 위한 대기 상태	현재 실행 중인 상태	실행을 종료한 상태

13. 라이브러리 활용하기

스레드

4 스레드 제어

- wait( )/notify( )/notifyAll( )

wait( )	notify( )/notifyAll( )
스레드 대기상태	스레드 대기상태 해제

- join( ) : 특정 스레드의 실행 종료를 기다리는 메소드
- sleep( ) : 인자로 지정된 시간동안 스레드의 실행을 잠시 멈춤
- interrupt( ) : 스레드의 상태를 종료시킴
- ThreadPool
  - 스레드를 효율적으로 사용할 수 있도록 관리하는 기술
- Semaphore
  - 제한된 자원을 효율적으로 활용하기 위한 기술
  - 실행할 수 있는 스레드 수를 제어하기 위한 카운터 제공