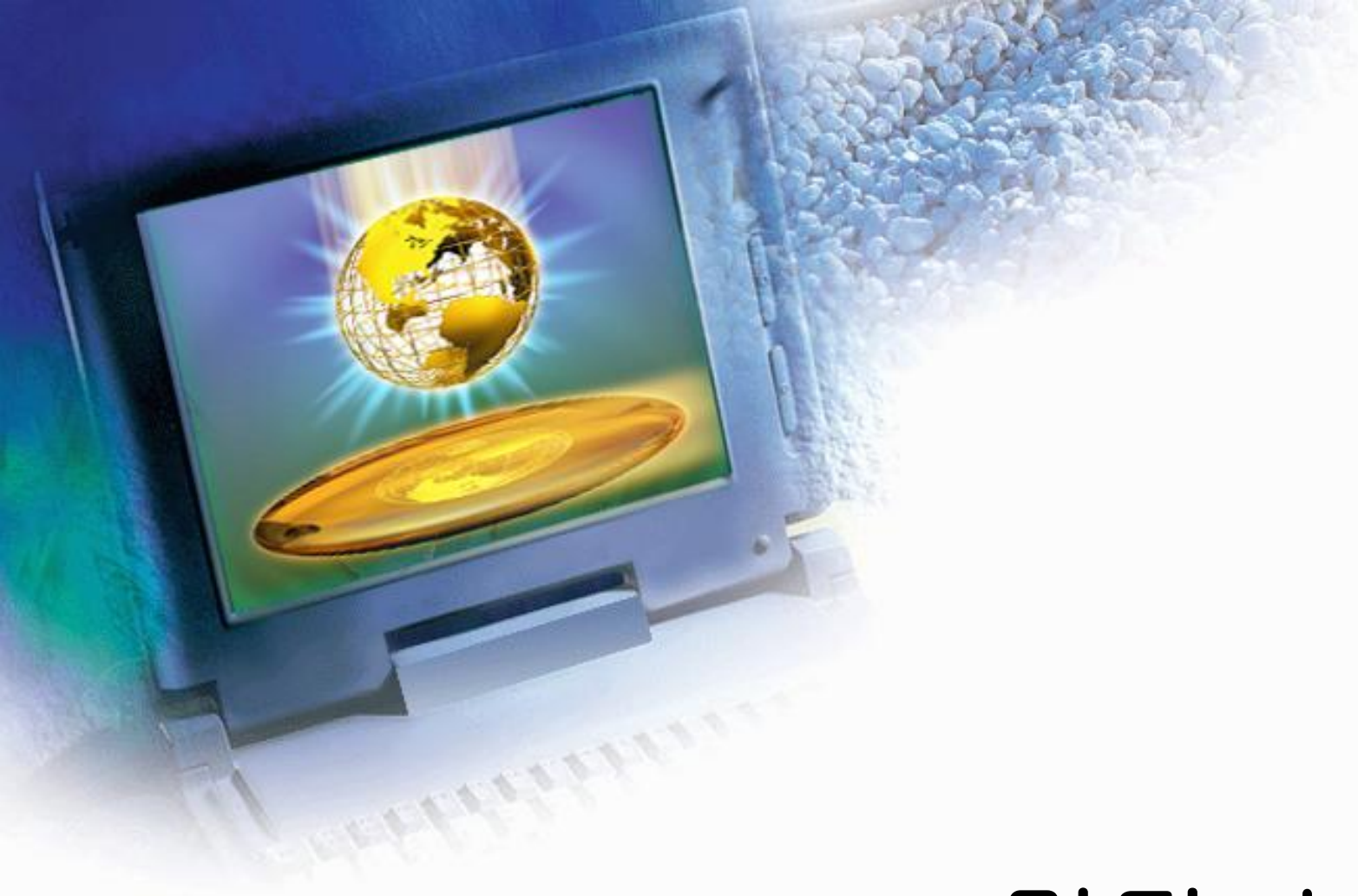


멀티 스레드

목표:

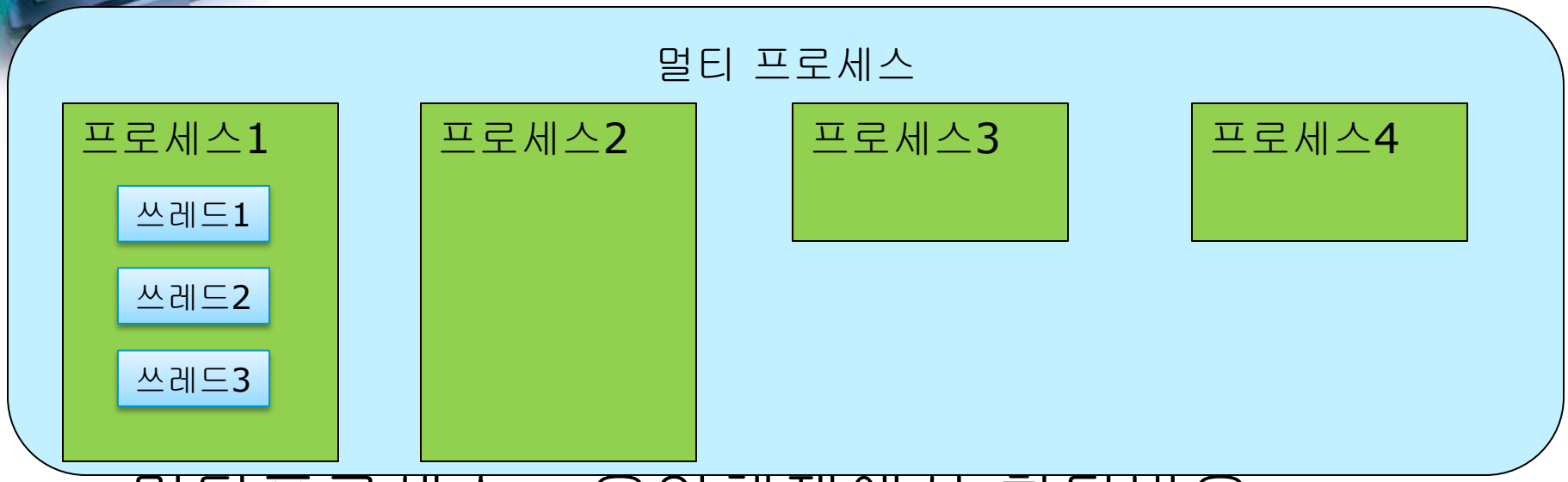
- 프로세스와 스레드를 구분할 수 있다.
- 멀티 태스킹 개념을 이해 한다.
- 자바에서 활용하는 **Thread**를 이해하고, 기본 코드와 확장 코드를 활용할 수 있다.
- **Thread**의 **life cycle**을 이해한다.

생각해봅시다

멀티 쓰레드 ?

- 운영체제
  - 프로그램(어플리케이션) 실행 시 프로세스가 진행된다.
  - 하나의 프로그램은 여러 다중 프로세스
    - 익스플로서를 두개 실행시, 두개의 프로세스가 생성
- 멀티 태스킹(multi tasking)
  - 두 가지 이상의 작업을 동시에 처리
  - 하나의 프로세스(프로그램)에서 두가지이상 작업 예를들어 미디어 플레이어와 메신저 실행 ➔ multi thread로 가능

멀티 프로세서와 멀티 쓰레드



- 멀티프로세스 : 운영체제에서 할당받은 메모리를 가지고 있기에 서로 독립적으로 처리
  - ex) 엑셀, 메신저
- 멀티쓰레드: 하나의 프로세스에서 실행되기에 오류에 영향을 미친다. ex) 메신저에서 파일업로드시 채팅에 영향을 미친다.

자바 코드를 통해 쓰레드 처리

- main()안에서 실행되는 내용은 실행흐름이 thread로 처리.
  - 싱글 쓰레드 애플리케이션
    - 프로세스(1) : 코드1 → 코드2 → 코드3 → 코드4
  - 멀티 쓰레드 애플리케이션
    - 프로세스(1)
      - 메드쓰레드　작업쓰레드2　　작업쓰레드3
      - 코드1
      - 코드2 → 코드2-1 → 코드2-2
      - 코드3　　→ 코드3-1 → 코드3-3
      - 코드4

Thread 생성과 실행

- Thread 클래스로부터 직접 생성
  - Runnable 인터페이스를 implements한 클래스를 Thread의 생성자 매개값으로 전달
    - class Task implements Runnable{}
    - Thread t1 = new Thread(new Task());
    - 다른 클래스를 상속해서 써야 하는 경우.
- Thread 하위 클래스로 부터 생성
  - Thread 클래스를 상속해서 바로 run()를 구현해서 실행
    - class Task2 extends Thread{}
    - Task2 t1 = new Task2();

Runnable 인터페이스 활용 확인

- ShoppingMall 접속하는 고객(Customer)를 Runnable interface implements해서 다음 같은 단계를 처리하는 내용을 구현하세요..
  - @@@님 @@ 단계를 처리했습니다.(1단계)
  - 단계: 로그인, 계정확인, 물건구매, 장바구니확인, 결재, 배송처리, 로그아웃(2단계)
  - @@@님 로그인 단계입니다.
- Customer 3명정도 접속시, Thread로 처리하는 내용 구현

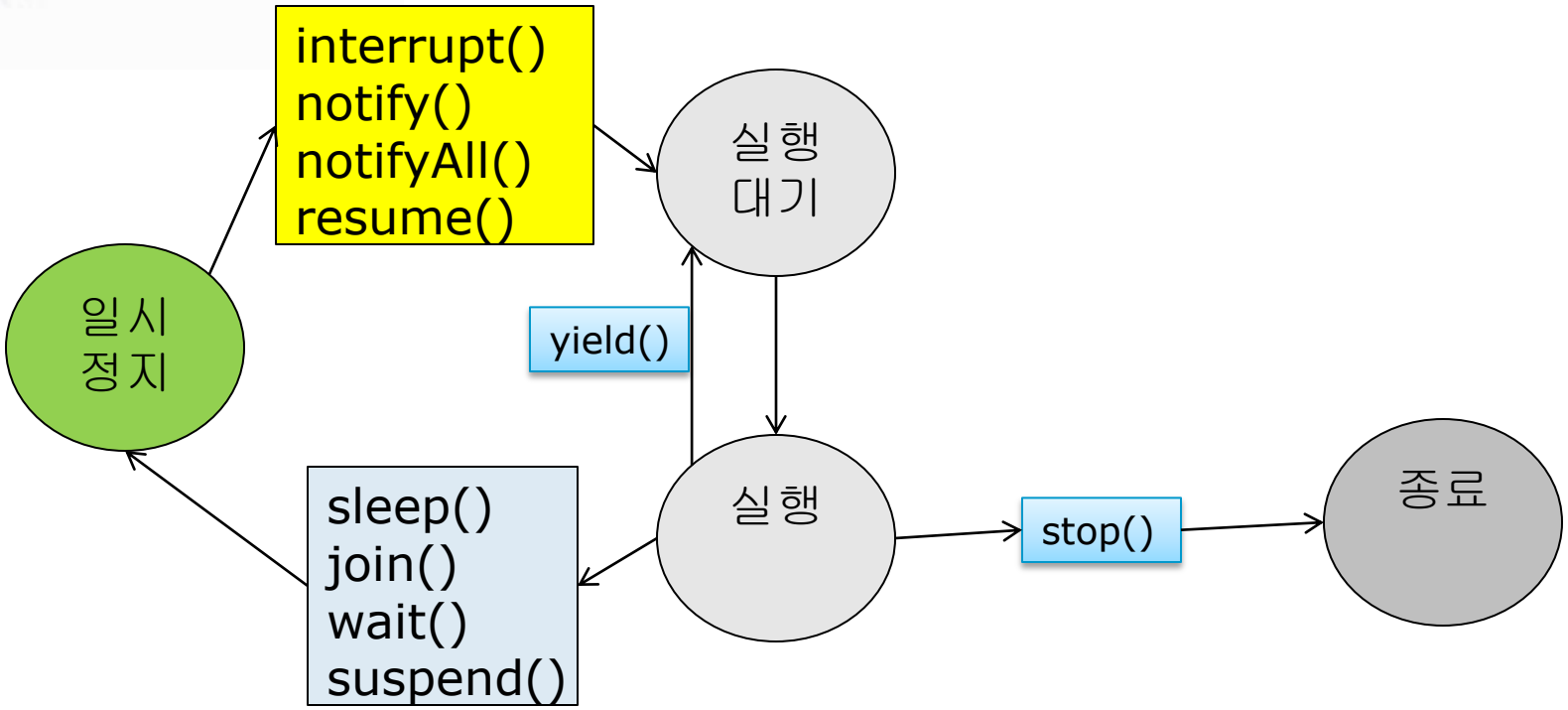
Thread 우선순위

- 멀티 쓰레드는 동시성 또는 병렬성으로 실행한다.
  - 동시성 : 멀티 작업을 위해서 cpu(하나의 코어)에 멀티 쓰레드가 번갈아가며 실행
  - 병렬성 : 멀티 작업을 cpu(멀티의 코어)에서 개별 쓰레드를 동시에 실행
- 특정한 경우에 우선 순위를 지정해서 처리해야 할 필요가 있는 경우
  - thread.setPriority(1~10)
    - Thread.MAX_PRIORITY : 10
  - 우선 순위가 높은 Thread는 실행할 기회를 더 많이 가지게 처리.

스레드의 상태(life cycle)

- 객체생성 : NEW
  - 스레드 객체가 생성, start()메서드가 호출X
  - ex) Thread t1 = new Thread();
- 실행대기 : RUNNABLE
  - 실행 상태로 언제든지 갈 수 있는 상태
  - ex) t1.start(); 실행 대기 상태에 있는 스레드 중에 스레드 스케줄링에 선택된 스레드가 비로서 CPU를 점유하고 run() 메소드를 실행한다.
  - 실행과 runnable한 내용이 반복적으로 처리된다.
- 일시정지
  - WAITING : 다른 쓰레드가 통지할 때까지 기다리는 상태
  - TIMED_WAITING : 주어진 시간 동안 기다리는 상태
  - BLOCKED : 사용하고자 하는 객체의 락이 풀릴 때까지 기다리는 상태
- 종료 : TERMINATED
  - 실행을 마친 상태

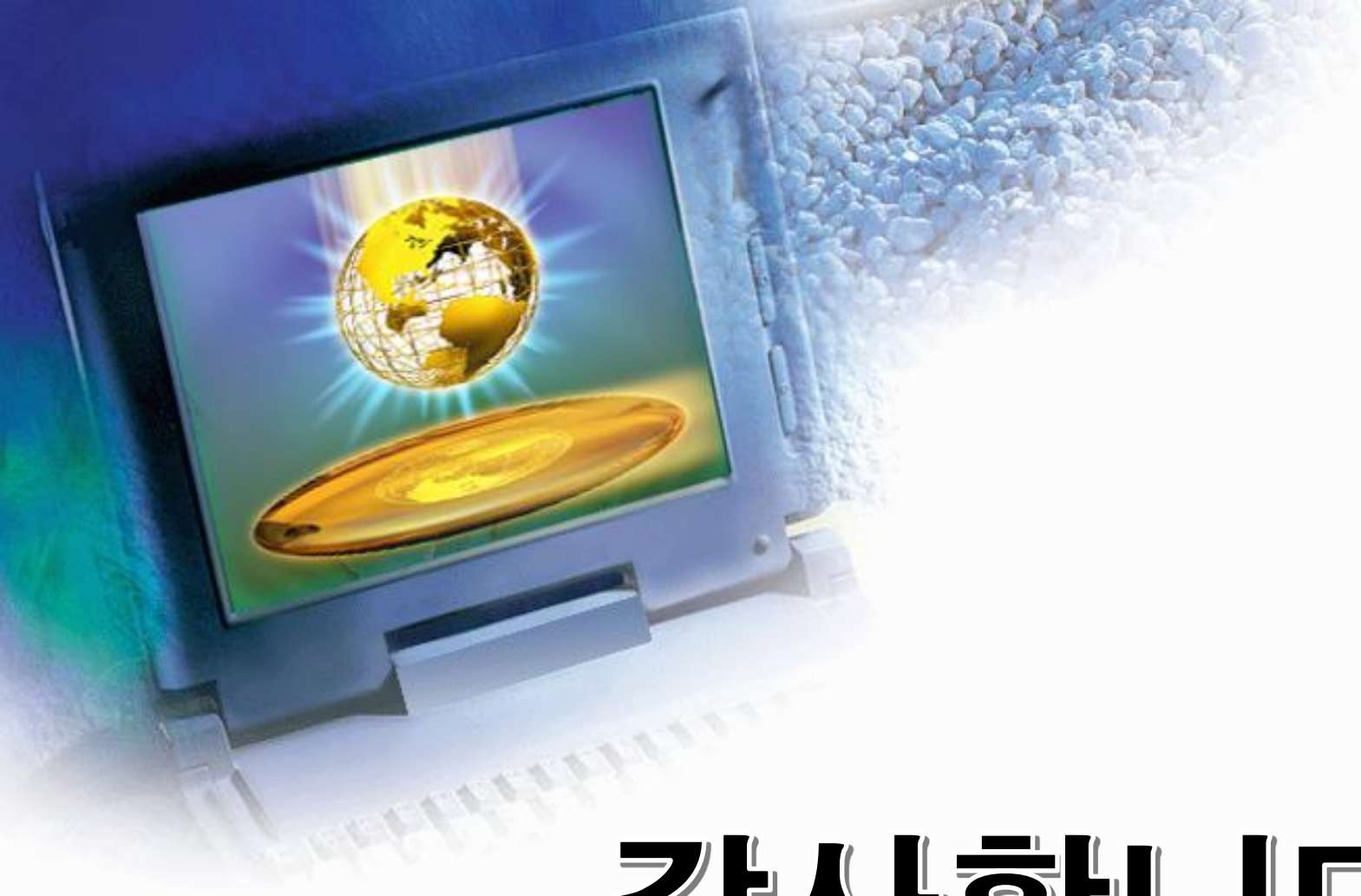
스레드 상태 제어


interrupt()
notify()
notifyAll()
resume()
실행
대기
일시
정지
yield()
sleep()
join()
wait()
suspend()
실행
stop()
종료

상태 제어 메서드

- interrupt() : 일시 정지 상태의 스레드에서 예외발생시켜, 실행 대기 상태로 가거나 종료
- notify(), notifyAll() : wait() 발생된 일시 정지 상태에 있는 스레드를 실행 대기 상태 만듦.
- resume() : suspend() 의해 일시 정지 상태에 있는 스레드를 실행 대기 상태로 만듦
- sleep(1/1000) : 주어진 시간, 동안 일시정지, 시간이 지나면 자동으로 실행 대기
- join(시간) : 호출한 스레드 일시정지. join() 메서드를 멤버로 가지는 스레드가 종료되거나, 시간이 지날때 실행대기
- yield():다른 스레드에 실행을 양보. 실행 대기 상태
- stop() : 스레드 즉시 종료

정리 및 확인하기



감사합니다 !