Java로 배우는 디자인패턴 입문
Chapter 5. Singleton
단 하나의 인스턴스

교재: 자바언어로배우는디자인패턴입문(개정판)/YukiHiroshi저/김윤정역/영진닷컴

덕성여자대학교 컴퓨터학과
최 승 훈

01. Singleton 패턴

- 프로그램 실행 시,
  - 하나의 클래스에 대한 인스턴스(객체)가 보통 여러 개 생성된다.

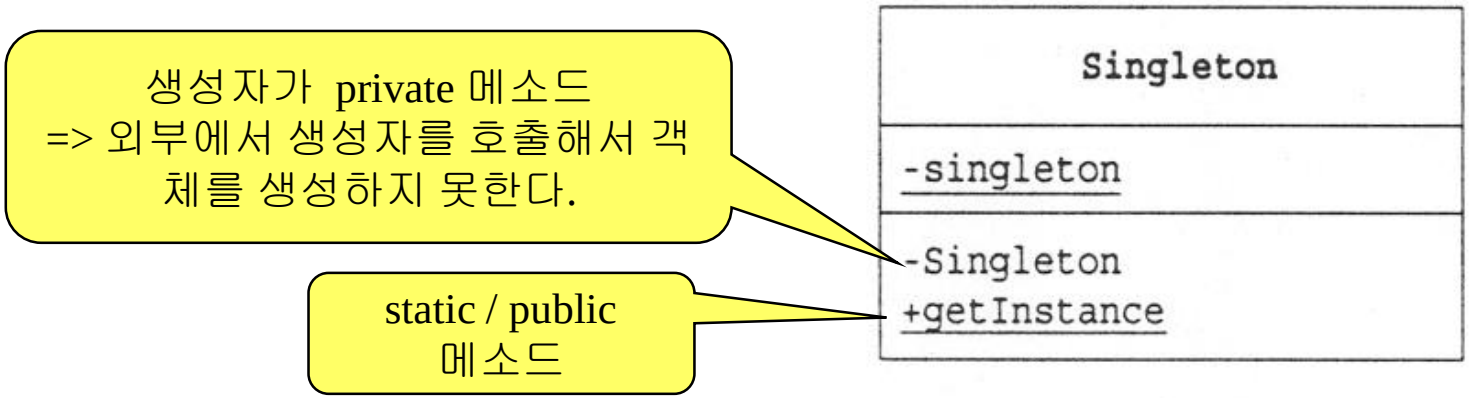
- 반드시 하나의 인스턴스만 생성되어야 하는 클래스도 있다.
  - 예: 컴퓨터 자체를 표현한 클래스, 윈도 시스템을 표현한 클래스

- 이 경우, 프로그래머가 new MyClass( )를 한번만 실행하면 된다.
  - 그러나, 반드시 1개의 인스턴스만 생성되도록 프로그램(코드)에 표현하고 싶다면,

    => Singleton 패턴을 사용한다.

02. 예제 프로그램

이름	해설
Singleton	인스턴스가 하나만 존재하는 클래스
Main	동작 테스트용 클래스



02. 예제 프로그램

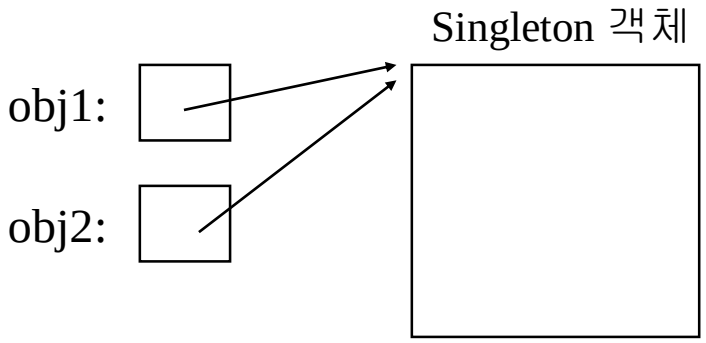
- Singleton 클래스 (리스트5-1)
  - private static singleton
    - Singleton 클래스의 인스턴스를 생성해서 가리킨다.
    - 정적 필드로서, Singleton 클래스를 로드할 때 ***한번만*** 실행된다.
    - private이므로, 외부에서 접근하지 못한다.
  - private Singleton( ) 생성자
    - private 메소드이므로, 외부에서 new Singleton( )을 호출하지 못함.
  - public static Singleton getInstance( )
    - Singleton 클래스의 유일한 하나의 인스턴스를 얻을 때 사용하는 메소드

02. 예제 프로그램

- Main 클래스 (리스트5-2)
  - Singleton.getInstance( )를 이용하여, Singleton 클래스의 객체를 얻어온다.
  -

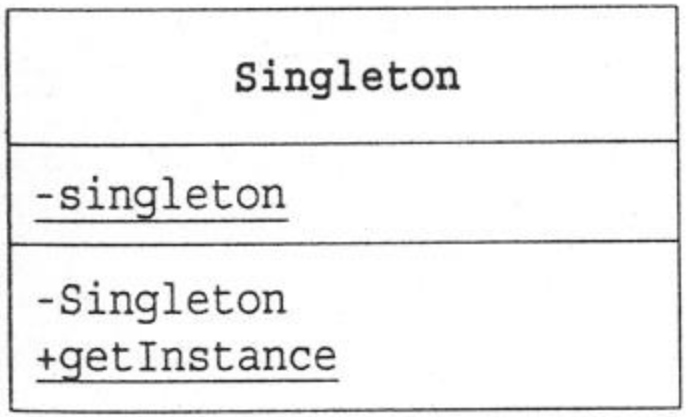
```
if(obj1 == obj2)
```

    - obj1변수와 ojb2 변수가 같은 객체를 참조(같은 객체의 주소를 저장)하는지 비교



03. 등장 역할

- Singleton의 역할



04. 독자의 사고를 넓혀주는 힌트

- 왜 제한할 필요가 있는가?
  - 인스턴스가 하나만 존재한다는 것이 보증되면, 인스턴스 상호 간에 영향을 주어 생각지 못한 버그가 발생할 가능성이 없어진다.

- 유일한 하나의 인스턴스는 언제 생성되는가
  - 프로그램 실행 후, 처음으로 Singleton.getInstance( ) 메소드가 호출되면, Singleton 클래스가 메모리에 로드되어 초기화되고, 이 때 static 필드인 singleton 필드가 초기화된다.

05. 관련 패턴

- Abstract Factory 패턴 (8장)
- Builder 패턴 (7장)
- Facade 패턴 (15장)
- Prototype 패턴 (6장)

06. 요약

- 하나의 인스턴스만 생성되어야 하는 클래스 구현 패턴

연습 문제

- 5-1. TicketMaker 클래스에 Singleton 패턴 적용하기
  - TicketMaker는 프로그램을 통틀어서 한 개의 인스턴스만 존재해야 한다.
    - 이유: 티켓번호는 고유해야 하므로.
  - 해답에서, getNextTicketNumber( )를 synchronized로 선언한 이유는?
    - 다수의 스레드가 동시에 이 메소드를 호출하면, 같은 값을 반환할 위험이 있기 때문
    - 한 스레드가 synchronized 메소드를 호출하여 실행 시, 다른 스레드가 그 메소드를 호출하면, 그 메소드 실행이 완료될 때까지 기다려야 한다.

연습 문제

- 5-2
  - 인스턴수의 개수가 3개로 한정되어 있는 클래스 Triple 만들기
    - 배열을 이용한다.

연습 문제

- 5-3. 리스트5-4(ch05.Q3.Singleton.java)가 Singleton 패턴이 되지 않는 이유는?

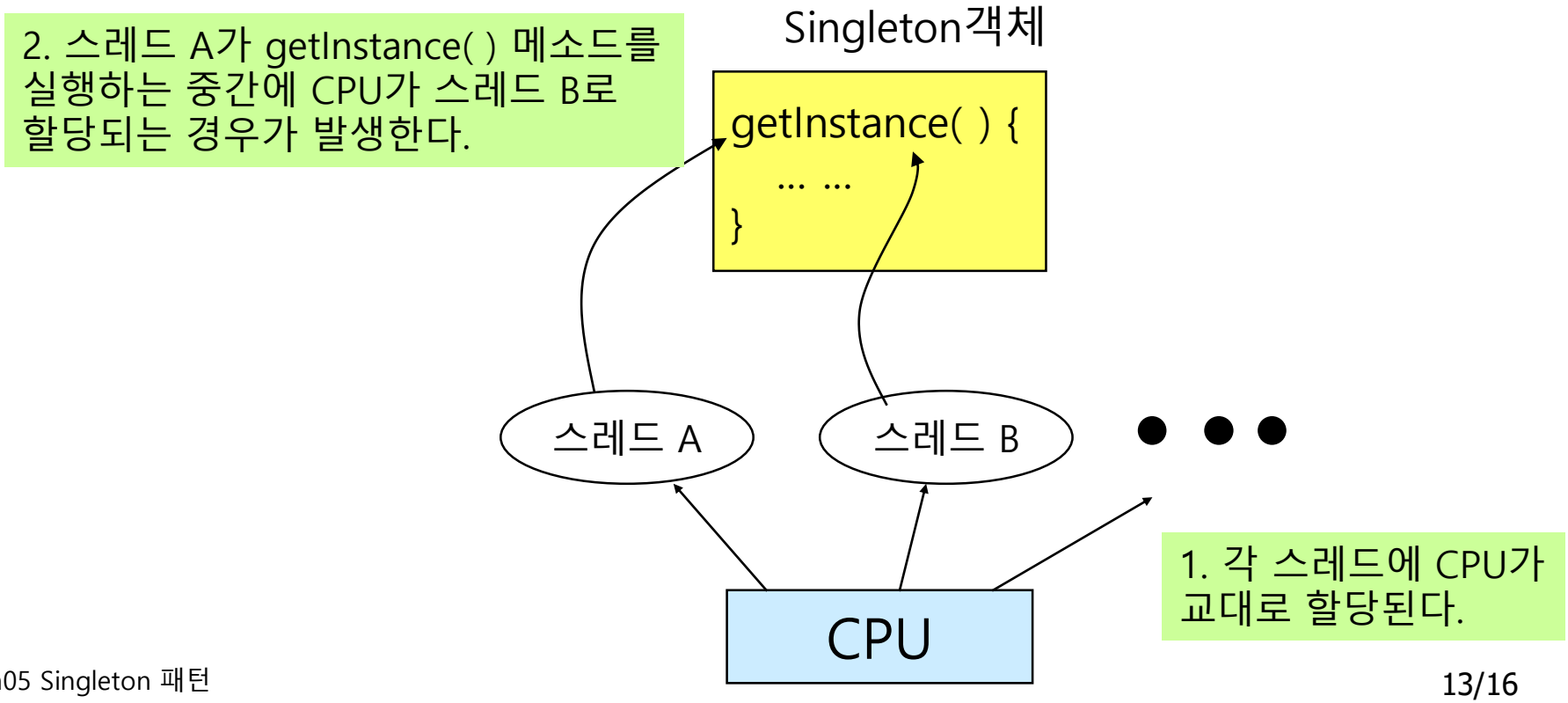
  - 구현 방식 설명

    - getInstance( ) 메소드에서, singleton 필드가 null 이면 Singleton 객체를 생성하고, null 이 아니면, 현재 있는 Singleton 객체를 반환한다.

    - null 은, 현재 Singleton 객체가 생성되었는지 안 되었는지를 나타내는 값으로 사용된다.

연습 문제

- 5-3(계속). 리스트5-4가 Singleton 패턴이 되지 않는 이유는?
  - Singleton 패턴이 안 되는 이유(다중 스레드 환경인 경우)



연습 문제

- 5-3(계속). 리스트5-4가 Singleton 패턴이 되지 않는 이유는?
  - Singleton 패턴이 안 되는 이유(다중 스레드 환경인 경우)
    - 두 스레드가 동시에 getInstance( )를 호출한 경우 첫 스레드가 (singleton == null) 인지 비교해서 '참'이므로 Singleton 객체를 생성하고, 그 객체를 singleton 변수에 할당하려고 한다.
    - 그런데, singleton 변수에 할당하기 직전에, 다른 스레드가 CPU를 할당받고 getInstance( )를 호출하여 (singleton == null) 을 비교하는 경우가 있다. 아직 singleton 변수에는 Singleton 객체가 할당되어 있지 않으므로, 비교 결과가 또 참이 되어 Singleton 객체를 또 생성하려고 한다.

연습 문제

- 5-3(계속). 리스트5-4가 Singleton 패턴이 되지 않는 이유는?
  - 결과적으로 두 개 이상의 Singleton 객체가 생성될 수 있다.
  - 예: 443~444페이지
    - ch05.A3_2
      - 이 예제는 다중 스레드 프로그램이 아니다.
        - Main이 Thread 클래스를 상속받지 않았음
      - 따라서, Singleton 클래스는 항상 하나의 인스턴스만 생성된다.
    - ch05.A3_1 (다중 프로그램이다)
      - Main이 Thread 클래스를 상속받음으로써 스레드가 된다.
      - 생성자 Singleton( )에서, 객체 생성시 일부러 속도를 늦추기 위해서 sleep 하고 CPU를 내 놓는다. => 두 개 이상의 인스턴스가 생성될 확률이 높아짐
      - Main을 실행해보면, 인스턴스가 여러 개 생기는 것을 알 수 있다.

연습 문제

- 5-3(계속). 리스트5-4가 Singleton 패턴이 되지 않는 이유는?
  - 해결책
    - getInstance( ) 메소드를 synchronized로 선언한다.
      - 동시에 두 개 이상의 스레드가 getInstance( ) 메소드 안으로 들어오지 않도록 막는다.