들어가기 전에

- 스터디 룰

1. 2주에 한번 강남 또는 판교 시간은 오후 1시~
2. 2주마다 돌아가면서 이끔이 역활을 맡아서 자료 및 장소 준비
3. 지각/결석은 벌금 1000원/5000원....장소 대여비 또는 음료,다과 등등으로 사용
4. 단, 부득이한 사정으로 참석 못 할 경우 3일전에 미리 공유
5. 주제에 집중해서 토론을 하며, 서로 말을 끊지 않고 끝까지 듣기
6. 스터디는 즐겁게 불편한 의견도 서로 수용

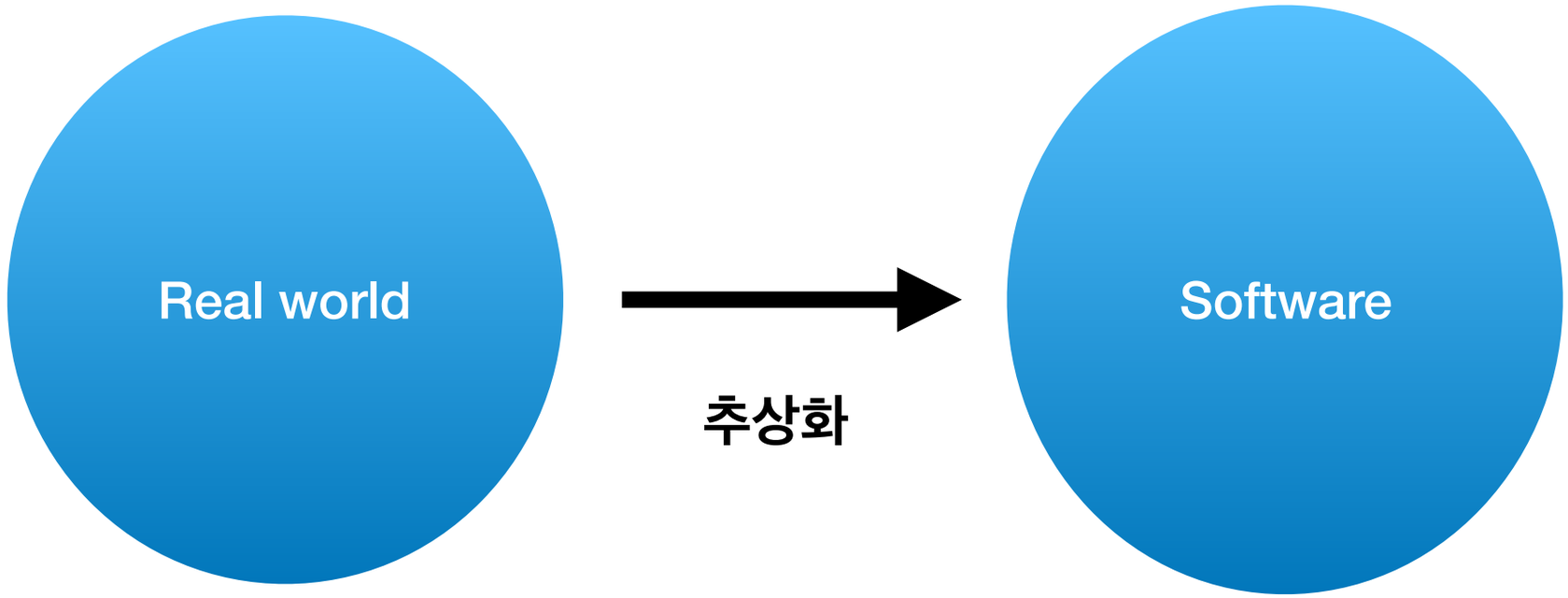
설계에서 중요한것은 무엇일까?

Design for Change

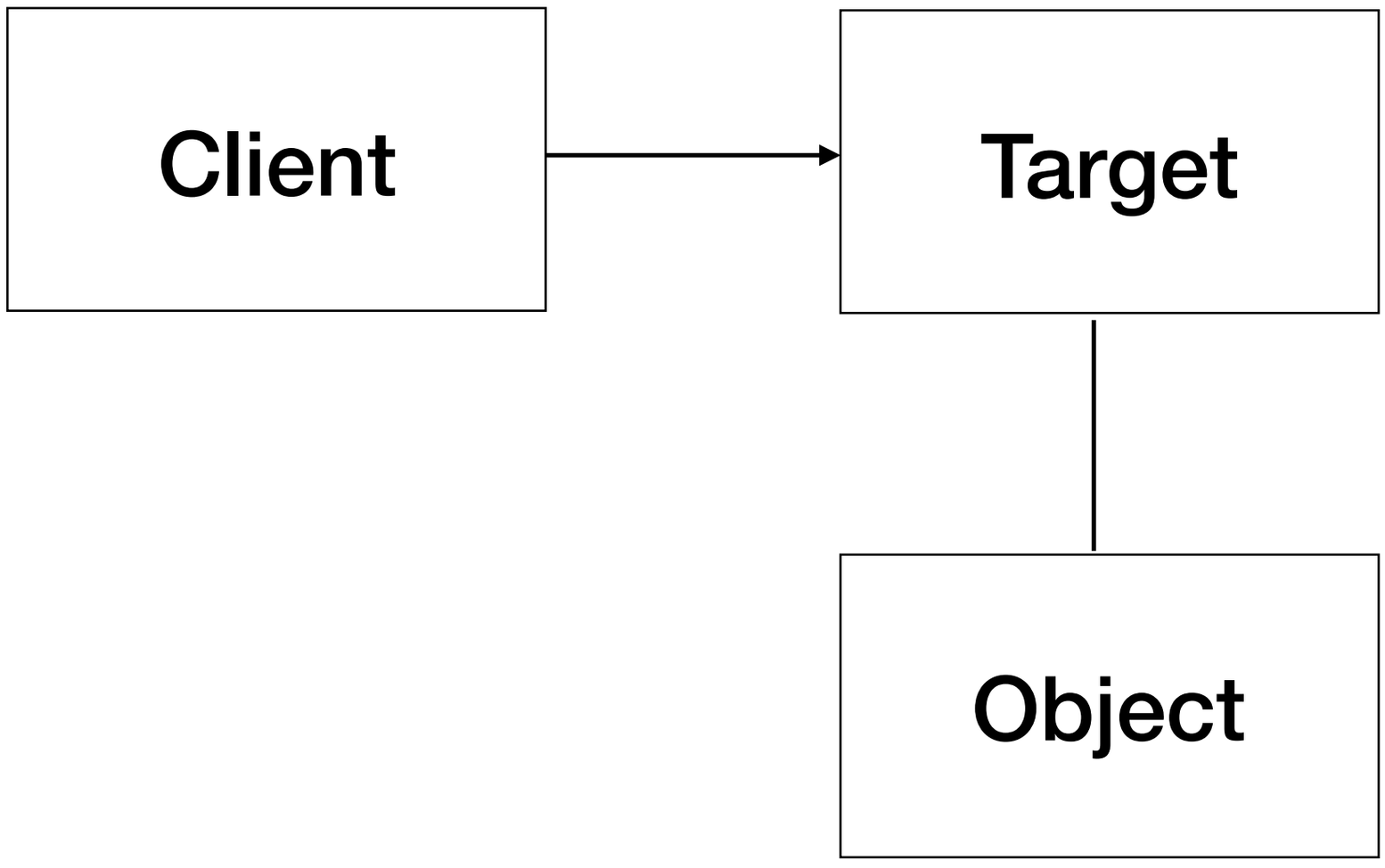
(변화를 수용할 수 있는 설계)

객체지향 설계

- 프로그래밍이란 무엇인가?



객체지향 설계

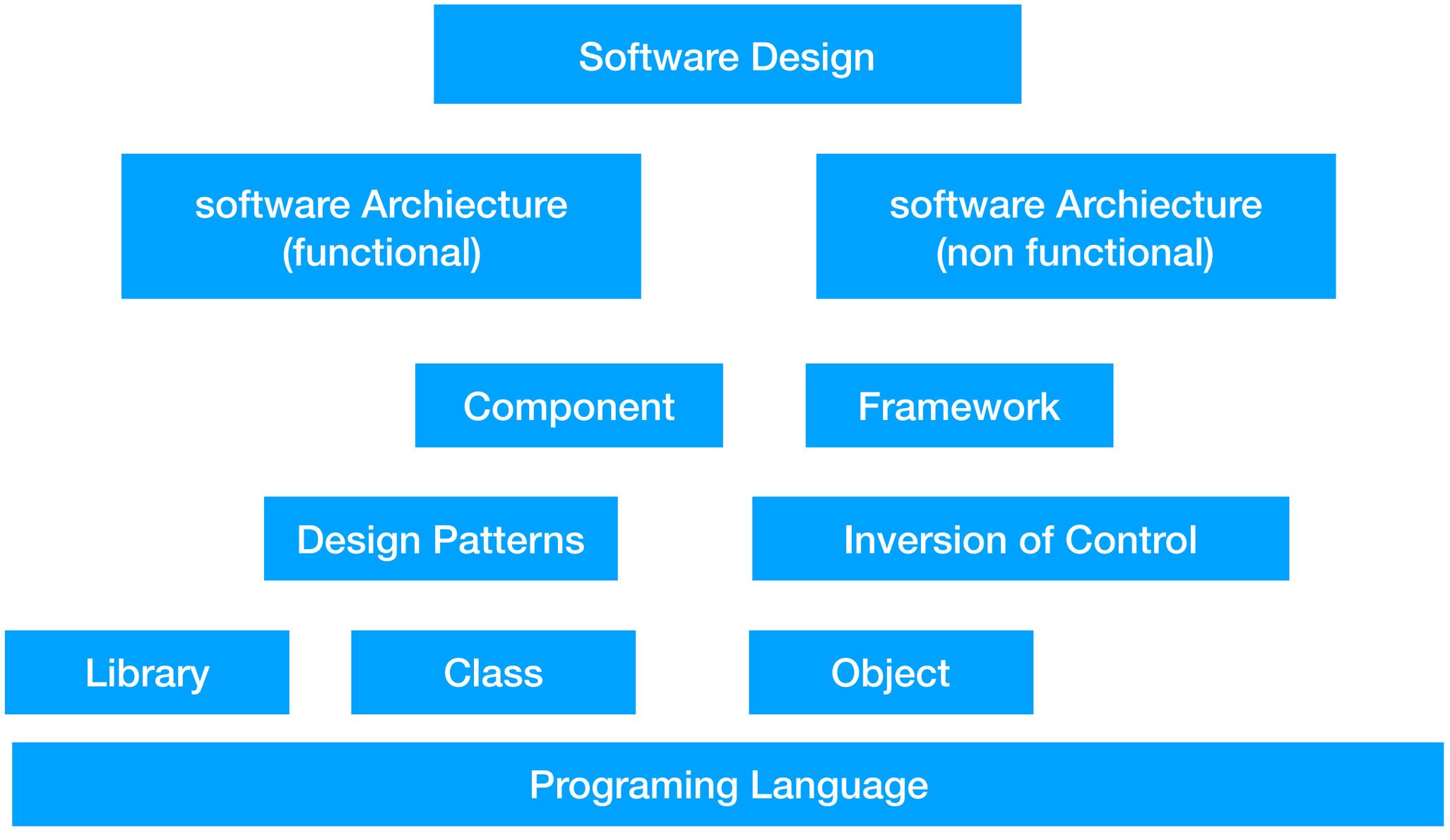


객체지향 설계

- 응집도(Cohesion), 결합도(Coupling)
    - 응집도 : 하나의 모듈이 하나의 기능을 온전히 담당하게하고 있는 정도
    - 결합도 : 모듈간에 서로 다른 기능이 상호의존하는 정도

객체지향 설계

- 우리가 스터디에서 공부하려고 하는 것



객체지향 설계

- 객체지향이란 무엇인가?

상호작용하는 객체의 집합

객체지향 설계

- 객체지향 프로그램의 목적은 결국은 협업하도록 구성된 객체들의 집합
    - 협업을 위한 메시지의 교환의 관계
- 좀더 상세하게
    - 캡슐화,다양화,상속 등을 이용해서 코드의 재사용성, 유지보수의 어려움을 감소 시킬수 있는 방법

객체지향 설계

- 객체지향을 지원하는 언어의 특징은?

    1. 추상화를 가능하게 하는 클래스와 객체를 제공
    2. 이미 존재하는 것으로부터 새로운 추상화를 만들어 내는것
    3. 런타임에 바인딩 할 수 있는 어떠한 형태를 제공

객체지향 설계

- 객체지향의 5가지 개념
  - 객체
  - 클래스
  - 캡슐화
  - 상속
  - 다형성

객체지향 설계

- 객체란?
    - 데이터를 가지고 있는 상태 (State)
    - 행위(Behavior)의 집합을 가지고 있는것
    - 객체를 구별할수 있도록 식별 (Identity)

객체지향 설계

- 적당한 객체를 찾는일

  즉, 객체에게 책임(Responsibility)을 결정하는 것이 설계의 출발점

객체지향 설계

- 클래스는 단지 객체 생성의 청사진일뿐
    - 단지 객체를 만드는데 사용되는 방법을 기술하는 도구
    - 객체가 ‘변수’라면 클래스는 ‘타입’이다.

객체지향 설계

- 캡술화란?
    - 객체의 속성과 행위를 하나로 묶고 실제 구현내용 일부를 외부에 감추어 은닉하는 것
    - 캡슐화는 인터페이스와 구현을 분리하는것에서 시작
    - 인터페이스는 객체의 외부에서 접근 가능
    - 구현은 객체 내부에서 접근 가능
    - 즉, 캡슐화를 통해서 정보를 은닉한다는 것은 데이터를 은닉하는 것이 아닌 구현을 은닉하는 것

객체지향 설계

- 상속
  - 어떤 클래스에서 더 특화된 클래스를 위한 행동을 제공
  - 즉 부모 클래스가 갖는 모든 데이터와 연산을 서브 클래스가 갖는것
  - 상속은 재사용 아닌 계층 구조의 표현에 초점

객체지향 설계

- 다형성
    - 한 행동을 여러방법으로 구현하고 상황에 따라 구현을 선택할 수 있도록 기능을 제공

객체지향 설계

- 설계의 품질을 높이기 위한 객체지향 설계 5원칙
    - 단일책임 원칙 (SRP: Single Responsibility Principle)
    - 개방폐쇠 원칙 (OCP: Open Close Principle)
    - 리스코프 대체 원칙 (LSP: Liskov Substitution Principle)
    - 인터페이스 분리 원칙 (ISP: Interface Segregation Principle)
    - 의존관계 역전 원칙 (DIP: Dependency Inversion Principle)

GoF의 디자인 패턴

Ch1.Intro

디자인 패턴이란?

- 패턴은 발명되는 것이 아니라 발견하는것
- 성공적인 프로젝트에서 여러 프로그래머들이 작성된 다양한 프로그램에서 반복적으로 발견 되는것
- 패턴은 문제를 해결하기 위한 해결방법을 일반화
- 디자인 패턴은 구현이 아닌 디자인!
- 즉 소프트웨어를 개발하기 위한 커뮤니케이션 도구

디자인 패턴이란?

- 디자인 패턴의 정의
    - 특정한 전후 관계에서 일반적 설계 문제를 해결하기 위해 상호교류하는 수정가능한 객체와 클래스들에 대한 설명
    - 패턴의 구성요소 : 패턴이름, 문제, 해법, 결과

디자인 패턴이란?

- 디자인 패턴을 고르는 방법
    - 패턴이 어떻게 문제를 해결하는지 파악
    - 패턴의 의도를 파악
    - 패턴들간의 관련성 파악
    - 재설계의 원인을 파악
    - 설계에서 가변성을 가져야 하는 부분이 무엇인지 파악

디자인 패턴 카탈로그

- 설계의 목적에 따른 분류
    - 생성 패턴 : 객체의 생성과 관련된 문제
    - 구조 패턴 : 프로그램의 구조적 구성 문제
    - 행동 패턴 : 런타임 시에 상호작용하는 객체들 간의 문제