! Design Pattern 시작

개괄

Design Pattern 이란? : 문맥 내에서, 문제를 해결할 수 있도록 제시된 디자인적인 해결책.

Context : Situation (문제를 당면한 상황, 좀 더 구체적인 예로, 주변 코드들과 기획)
Problem : 해결해야하는 문제, 성취할 요소.
Solution : General "Design" => Design Pattern : 해당 문제를 해결할 수 있는 방법.

> *내가 생각했을 때 가장 중점적인 내용.*
>
> - *일단 Design Pattern 을 배웠다 치자. **어떻게 써먹을까?***
>   - ***Context & Problem** : 문제 상황 및 문제를 이루고 있는 문맥을 인지.*
>   - ***Actual Pattern** : 해당 문제 상황을 해결하기 위해, 어떤 패턴을 사용할지 결정 및 구현.*
>   - *위의 까지 하면, 사용에 있어서는 문제가 없을 것 같은데, 배울 때는 "**Goal - 해당 구조를 구현함으로써 얻을 수 있는 것**"까지 정리하여 두는 게 좋을 듯함. => 이는 결국 Context & Problem 을 해결할 수 있다는 것이 중점적이므로, Goal 에서 얻을 수 있는 이점이 별도로 크게 존재하는 것이 아닐 경우, 합쳐도 되지 않을까 함.*

이는 !!! 개발 (D-I-T) 과정 !!! 에서, Module 결정 및 설계 과정에서 아예 직접적으로 많이 써먹게 되는 지식이 아닐까 함.

> *또한, Pattern 자체는 추상적인 구조이자, 개념이자, 경험임. 이를 직접적으로 구현하는 것, 구현할 때 지칭하는 용어와 / Design 패턴에서의 구조의 이름을 별개로서 생각하는 것이 좋음.*

Design Pattern Categories

- Creational : 생성 방법에 대한 결정. 내용을 감추고, 어떻게 생성할지의 방법.
- Behavior : Object 가 상호작용하고, 의존성을 분산시키는 방법.
- Structural : Class 와 Object 사이의 구조, 계층, 구성을 설정하는 방법.