Flyweight Pattern

결론 및 정리

결국 대량의 Object 가 존재할 때, 그 Object 가 지니고 있는 공동의 데이터의 경우, 따로 해당 데이터만을 저장하는 Class 를 만들어 공유시키는 것이 이 Pattern 의 중점.

정의

Flyweight Pattern은 Structural Pattern 중 하나이다.
많은 수의 유사한 객체가 존재한다 하자. 이 때 해당 객체 내에 사용되는 유사한 정보를 공유하는 다른 Class 를 둠으로써, 메모리 사용량을 치소화하는 pattern 이다.
이 패턴은 주로 세부 정보가 많은 객체에서 공통 데이터를 분리하여 공유하도록 설계된다.

- **Flyweight** : 모든 ConcreteFlyweight 객체의 공통 인터페이스.
- **ConcreteFlyweight** : Flyweight 인터페이스를 구현하며, "공유"될 수 있는 데이터를 포함하는 객체이다.
  repeating 되는, intrinsic data 를 가지고 있다.
- **FlyweightFactory** : ConcreteFlyweight 객체를 생성하고 관리하는 클래스. 클라이언트의 요청에 따라 이미 생성된 객체를 반환하거나 새로운 객체를 생성한다.

구현

1. Flyweight 인터페이스를 정의한다.
2. ConcreteFlyweight 클래스에서 Flyweight 인터페이스를 구현하며, 공유될 수 있는 데이터(내부 상태)를 저장한다.
3. FlyweightFactory 클래스를 사용하여 ConcreteFlyweight 객체를 생성하고 관리한다.
4. 클라이언트는 FlyweightFactory를 통해 필요한 Flyweight 객체를 얻는다.

Context & Problem & Merit

- 많은 수의 유사한 객체를 생성해야 하며, 이로 인해 메모리 사용량이 크게 증가할 때 Flyweight Pattern을 고려한다.
- 객체의 대부분의 상태 정보가 공유될 수 있을 때 유용하다.

1. **메모리 절약**: 많은 수의 유사한 객체를 공유함으로써 메모리 사용량을 크게 줄일 수 있다.
2. **성능 향상**: 객체 생성 비용이 큰 경우, 객체를 재사용함으로써 성능을 향상시킬 수 있다.
3. **구조화**: 객체의 내부 상태와 외부 상태를 분리하여, 시스템의 구조를 더 명확하게 만들 수 있다.