1. 인터페이스

1.1 인터페이스의 개념

① 작업 명세서(작업 지시서): 미리 만들 프로그램의 객체를 구현해 놓은 것
- 실제 구현된 것이 전혀 없는 기본 설계도. (알맹이 없는 껍데기)
- 추상 클래스(미완성 설계도)보다 추상화 정도가 높다.
- 인스턴스를 생성할 수 없고, 클래스 작성에 도움을 줄 목적으로 사용된다.
- 미리 정해진 규칙에 맞게 구현하도록 표준을 제시하는 데 사용된다.
- 추상 메서드와 상수 만을 멤버로 가질 수 있다.

② 다형성을 가능하게 한다. (하나의 객체를 다양하게 많은 type으로 만들 수 있다.)

③ 객체의 부속품 화 -다양한 객체를 제품의 부속품처럼 개발자 마음대로 변경 할 수 있다.

④ 사용법은 어렵지 않지만, 실제 개발에 적용시키기는 쉽지 않다.

⑤ 인터페이스를 공부하는데 가장 좋은 방법은 패턴이나 프레임워크(ex. Spring)를 통해 습득하는 것.

⑥ 객체와 객체 간의 소통 수단으로 이용.

1.2 인터페이스의 문법 활용

① 'class'대신 'interface' 예약어를 사용한다는 점에서 클래스와 유사하다.

② 실제 구현된 기능 없이 추상 메서드와 상수만이 존재

```
public interface 인터페이스이름 {
        public static final 타입 상수 이름 = 값;
        public abstract 메서드 이름(매개변수 목록);
}
```

- 모든 멤버 변수는 public static final이어야하며 이를 생략할 수 있다.'
- 모든 메서드는 public abstract 이어야 하며, 이를 생략할 수 있다.

③ private는 불가 - 상수나 메서드를 만들 때 private 접근 제한 자는 불가

④ 추상화 - 메서드는 무조건 추상 메서드만 존재.

⑤ 변수 타입 - 인터페이스는 객체를 생성할 수 없다. 다만, 변수 타입으로만 사용.
(예외, 익명 구현 객체만이 가능은 하다. 안드로이드에서 주로)

⑥ 구현은 Implement 되는 클래스에서 한다.

2020-12-09