인터페이스 사용

인터페이스 사용

❖ 인터페이스에 구현 객체를 대입하는 방법

```
인터페이스 변수;
변수 = 구현객체;
```

```
인터페이스 변수 = 구현객체;
```

```
RemoteControl rc;
rc = new Television();
rc = new Audio();
```

인터페이스 사용

❖ 인터페이스에 구현 객체를 대입하는 방법

```
public class MyClass {
    //필드
    RemoteControl rc = new Television();

    //생성자
    MyClass(RemoteControl rc) {
      this.rc = rc;
    }

    //메소드
    void methodA() {
      //로컬 변수
      RemoteControl rc = new Audio();
    }

    void methodB(RemoteControl rc) { … }
}
```

```
MyClass mc = new MyClass(new Television());
```

```
mc.methodB(new Audio());
```

인터페이스 사용

❖ 추상 메소드 사용

```
RemoteControl rc = new Television();
rc.turnOn();    → Television 의 turnOn() 실행
rc.turnOff();   → Television 의 turnOff() 실행
```

인터페이스 사용

❖ 인터페이스: RemoteControlExample.java

```
public class RemoteControlExample {
    public static void main(String[] args) {
        RemoteControl rc = null;      // 인터페이스 변수 선언
        rc = new Television(); // 구현체(Television)를 인터페이스 타입에 대입
        rc.turnOn();           // 인터페이스 메서드 호출
        rc.turnOff();

        rc = new Audio();          // 구현체(Television)를 인터페이스 타입에 대입

        rc.turnOn();               // 인터페이스 메서드 호출
        rc.turnOff();

        SmartTelevision st = new SmartTelevision();
        RemoteControl rc2 = st;
        rc2.turnOn();
        Searchable sc = st;
        sc.search("인터페이스");
    }
}
```

인터페이스 사용

❖ 디폴트 메소드 사용

- 인터페이스만으로는 사용 불가
  - 구현 객체가 인터페이스에 대입되어야 호출할 수 있는 인스턴스 메소드

- 모든 구현 객체가 가지고 있는 기본 메소드로 사용
  - 필요에 따라 구현 클래스가 디폴트 메소드 재정의해 사용

인터페이스 사용

❖ 구현 클래스: Audio.java

```
public class Audio implements RemoteControl {
    // 필드
    private int volume;
    private boolean mute;

    // turnOn() 추상 메소드의 실체 메소드
    public void turnOn() {
        System.out.println("Audio를 켭니다.");
    }

    // turnOff() 추상 메소드의 실체 메소드
    public void turnOff() {
        System.out.println("Audio를 끕니다.");
    }
```

❖ 구현 클래스: Audio.java

```
    // setVolume() 추상 메소드의 실체 메소드
    public void setVolume(int volume) {
        if (volume > RemoteControl.MAX_VOLUME) {
            this.volume = RemoteControl.MAX_VOLUME;
        } else if (volume < RemoteControl.MIN_VOLUME) {
            this.volume = RemoteControl.MIN_VOLUME;
        } else {
            this.volume = volume;
        }
        System.out.println("현재 Audio 볼륨: " + volume);
    }

    @Override
    public void setMute(boolean mute) {
        this.mute = mute;
        if (mute) {
            System.out.println("Audio 무음 처리합니다.");
        } else {
            System.out.println("Audio 무음 해제합니다.");
        }
    }
}
```

인터페이스 사용

❖ 디폴트 메서드 사용: RemoteControlExample.java

```
public class RemoteControlExample {
    public static void main(String[] args) {
        RemoteControl rc = null;

        rc = new Television();
        rc.turnOn();
        rc.setMute(true);

        rc = new Audio();
        rc.turnOn();
        rc.setMute(true);
    }
}
```

인터페이스 사용

❖ 정적 메소드 사용

- 인터페이스로 바로 호출 가능

```
public class RemoteControlExample {
    public static void main(String[] args) {
        RemoteControl.changeBattery();
    }
}
```