KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

Prototype - 복사해서 인스턴스를 만든다

[KB] IT's Your Life

KB 국민은행

Prototype 패턴

Prototype 패턴

- 클래스 이름을 지정하지 않고 인스턴스를 생성하고 싶을 때

→ 클래스로부터 인스턴스를 복사해서 새 인스턴스를 생성

Prototype 패턴

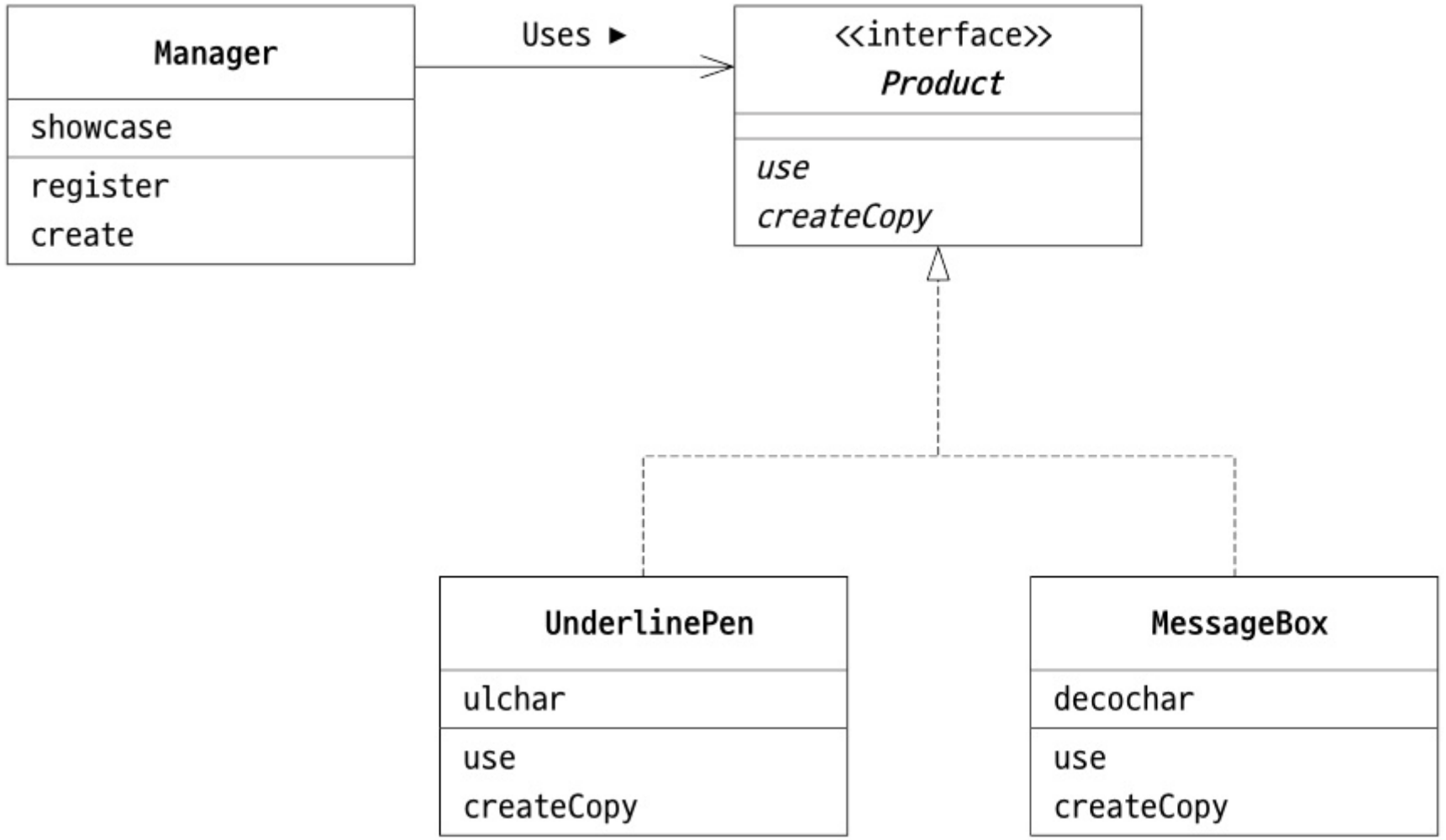
예제 프로그램

- 문자열을 테두리로 감싸서 표시하거나 밑줄을 그어 표시함

패키지	이름	설명
framework	Product	추상 메소드 use와 createCopy가 선언되어 있는 인터페이스
framework	Manager	createCopy를 사용하여 인스턴스를 복제하는 클래스
이름 없음	MessageBox	문자열을 테두리로 감싸서 표시하는 클래스로 use와 createCopy를 구현
이름 없음	UnderlinePen	문자열에 밑줄을 그어 표시하는 클래스로 use와 createCopy를 구현
이름 없음	Main	동작 테스트용 클래스

Prototype 패턴

예제 프로그램의 클래스 다이어그램



framework/Product.java

```
public interface Product extends Cloneable{
    void use(String s);
    Product createCopy();
}
```

framework/Manager.java

```
public class Manager {
    private Map<String, Product> showcase = new HashMap<>();

    public void register(String name, Product prototype) {
        showcase.put(name, prototype);
    }

    public Product create(String prototypeName) {
        Product p = showcase.get(prototypeName);
        return p.createCopy();
    }
}
```

MessageBox.java

```
public class MessageBox implements Product {
    private char decochar;

    public MessageBox(char decochar) {
        this.decochar = decochar;
    }

    @Override
    public void use(String s) {
        int decolen = 1 + s.length() + 1;
        for(int i = 0; i < decolen; i++) {
            System.out.print(decochar);
        }
        System.out.println();
        System.out.println(decochar + s + decochar);
        for(int i = 0; i < decolen; i++) {
            System.out.print(decochar);
        }
        System.out.println();
    }
```

MessageBox.java

```
    @Override
    public Product createCopy() {
        Product p = null;

        try {
            p = (Product) clone();
        } catch (CloneNotSupportedException e) {
            e.printStackTrace();
        }
        return p;
    }
}
```

UnderlinePen.java

```
public class UnderlinePen implements Product {
    private char ulchar;

    public UnderlinePen(char ulchar) {
        this.ulchar = ulchar;
    }

    @Override
    public void use(String s) {
        int ulen = s.length();
        System.out.println(s);
        for(int i = 0; i < ulen; i++) {
            System.out.print(ulchar);
        }
        System.out.println();
    }
```

Prototype 패턴

UnderlinePen.java

```
    @Override
    public Product createCopy() {
        Product p = null;

        try {
            p = (Product) clone();
        } catch (CloneNotSupportedException e) {
            e.printStackTrace();
        }
        return p;
    }
}
```

Main.java

```
public class Main {
    public static void main(String[] args) {
        // 준비
        Manager manager = new Manager();

        UnderlinePen upen = new UnderlinePen('-');
        MessageBox mbox = new MessageBox('*');
        MessageBox sbox = new MessageBox('/');

        // 등록
        manager.register("strong message", upen);
        manager.register("warning box", mbox);
        manager.register("slash box", sbox);
```

Prototype 패턴

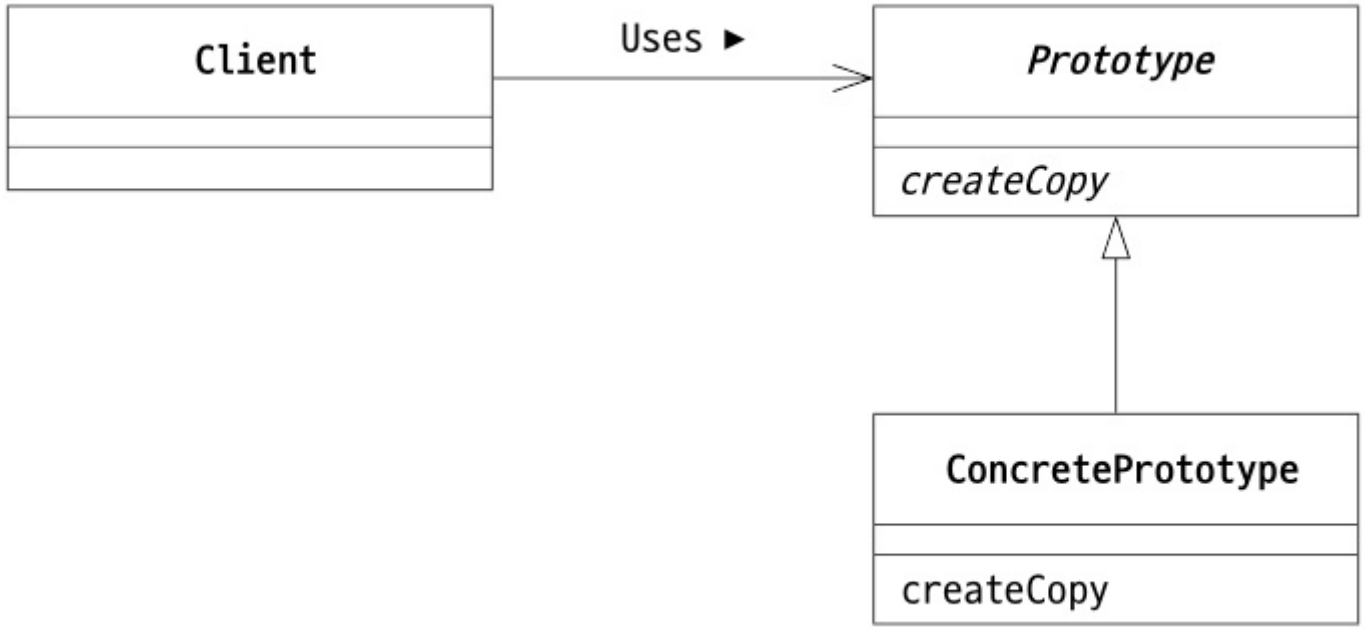
Main.java

```
        // 생성과 사용
        Product p1 = manager.create("strong message");
        p1.use("Hello, world.");

        Product p2 = manager.create("warning box");
        p2.use("Hello, world.");

        Product p3 = manager.create("slash box");
        p3.use("Hello, world.");

    }
}
```

```
Hello, world.
-------------
***************
*Hello, world.*
***************
///////////////
/Hello, world./
///////////////
```

Prototype 패턴

Prototype 패턴의 클래스 다이어그램



Prototype 패턴

Prototype 패턴을 사용하는 이유

- 종류가 너무 많아서 클래스로 정리할 수 없을 경우
- 클래스로부터 인스턴스 생성이 어려운 경우
- 프레임워크와 생성하는 인스턴스를 분리하고 싶은 경우

클래스 이름을 통해 인스턴스를 얻는 방법?

- 소스 코드안에 이용할 클래스 이름을 이용해 직접 생성시

→ 클래스와 분리해서 재사용할 수 없음(수정 발생)

- 부품으로서의 재사용 → 코드를 수정하지 않음