KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

Decorator - 장식틀과 내용물을 동일시한다

[KB] IT's Your Life

KB 국민은행

Decorator 패턴

- 스펀지 케이크가 있다고 가정
  - 중심이되는 객체
  - 크림을 바르면 크림 케이크가 됨
  - 딸기를 얹으면 딸기 크림 케이크 됨
  - 납작한 초콜릿을 올리고, 기타 장식을 붙이면 생일 케이크가 완성

- Decorator 패턴
  - 기본 중심 객체에 점점 장식을 더해 가는 디자인 패턴

예제 프로그램

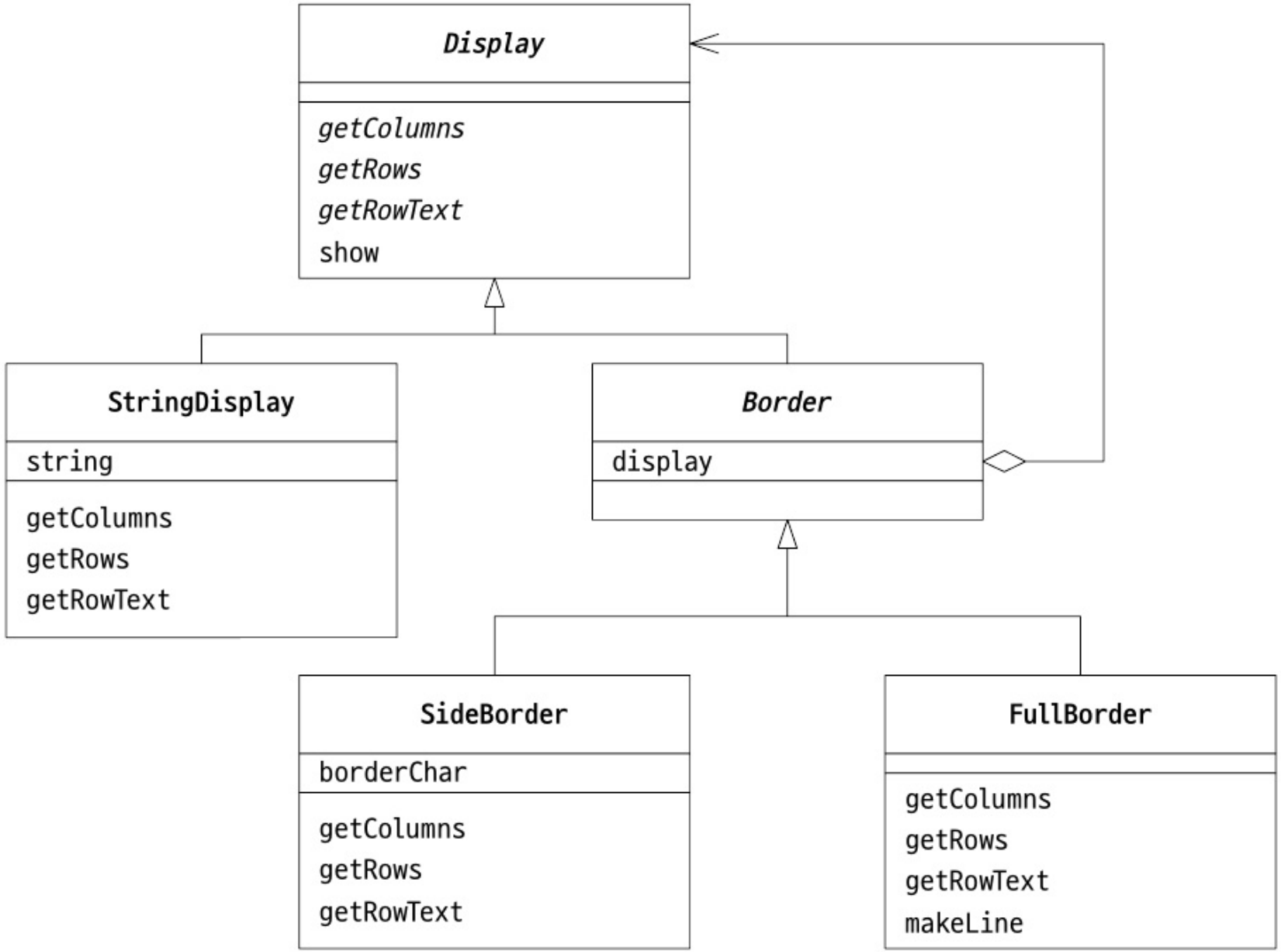
- 문자열 주위에 장식 틀을 붙여 표시하는 것
- Hello, world 라는 문자열에 장식틀을 붙인 예

```
+------------+
|Hello, world.|
+------------+
```

이름	설명
Display	문자열 표시용 추상 클래스
StringDisplay	1행으로 구성된 문자열 표시용 클래스
Border	'장식틀'을 나타내는 추상 클래스
SideBorder	좌우에만 장식틀을 붙이는 클래스
FullBorder	상하좌우에 장식틀을 붙이는 클래스
Main	동작 테스트용 클래스

예제 프로그램

- 예제 프로그램의 클래스 다



Display.java

```
public abstract class Display {

    public abstract int getColumns(); // 가로 문자 수를 얻는다.
    public abstract int getRows();    // 세로 행수를 얻는다.
    public abstract String getRowText(int row); // row행째 문자열을 얻는다.


    // 모든 행을 표시한다
    public void show() {
        for(int i =0; i < getRows(); i++) {
            System.out.println(getRowText(i));
        }
    }

}
```

StringDisplay.java

```
public class StringDisplay extends Display{

    private String string;  // 표시 문자열

    public StringDisplay(String string) {
        this.string = string;
    }

    @Override
    public int getColumns() {
        return string.length();
    }
    @Override
    public int getRows() {
        return 1;
    }

    @Override
    public String getRowText(int row) {
        if(row != 0) {
            throw new IndexOutOfBoundsException();
        }
        return string;
    }
}
```

예제 프로그램

Border.java

```
public abstract class Border extends Display{
    protected Display display;  // 이 장식들이 감싸는 '내용물'

    protected Border(Display display) { // 인스턴스 생성 시 '내용물'을 인수로 지정
        this.display = display;
    }
}
```

SideBorder.java

```
public class SideBorder extends Border{
    private char borderChar;    // 장식 문자

    public SideBorder(Display display, char borderChar) {
        super(display);
        this.borderChar = borderChar;
    }

    @Override
    public int getColumns() {
        // 문자 수는 내용물의 양쪽에 장식 문자만큼 더한 것
        return 1 + display.getColumns() + 1;
    }

    @Override
    public int getRows() {
        // 행수는 내용물의 행수와 같다
        return display.getRows();
    }

    @Override
    public String getRowText(int row) {
        // 지정 행의 내용은 내용물의 지정 행 양쪽에 장식 문자를 붙인 것
        return borderChar + display.getRowText(row) + borderChar;
    }
}
```

FullBorder.java

```
public class FullBorder extends Border {
    public FullBorder(Display display) {
        super(display);
    }

    @Override
    public int getColumns() {
        // 문자 수는 내용물 양쪽에 좌우 장식 문자만큼 더한 것
        return 1 + display.getColumns() + 1;
    }

    @Override
    public int getRows() {
        // 행수는 내용물의 행수에 상하 장식을 문자만큼 더한 것
        return 1 + display.getRows() + 1;
    }
```

FullBorder.java

```
    @Override
    public String getRowText(int row) {
        if (row == 0) { // 상단 테두리
            return "+" + makeLine('-', display.getColumns()) + "+";
        } else if (row == display.getRows() + 1) { // 하단 테두리
            return "+" + makeLine('-', display.getColumns()) + "+";
        } else {    // 기타
            return "¦" + display.getRowText(row - 1) + "¦";
        }
    }

    // 문자 ch로 count 수만큼 연속한 문자열을 만든다
    private String makeLine(char ch, int count) {
        StringBuilder line = new StringBuilder();
        for (int i = 0; i < count; i++) {
            line.append(ch);
        }
        return line.toString();
    }

}
```

예제 프로그램

Main.java

```
public class Main {
    public static void main(String[] args) {
        Display b1 = new StringDisplay("Hello, world.");
        Display b2 = new SideBorder(b1, '#');
        Display b3 = new FullBorder(b2);

        b1.show();
        b2.show();
        b3.show();

        Display b4 = new SideBorder(
          new FullBorder(
                  new FullBorder(
                          new SideBorder(
                                  new FullBorder(
                                    new StringDisplay("Hello, World.")
                                  ), '*'
                          )
                  )
          ), '/'
        );

        b4.show();
    }
}
```


```
Hello, world.                ← b1.show() 표시
#Hello, world.#              ← b2.show() 표시
+--------------+             ← b3.show() 표시
|#Hello, world.#|
+--------------+
/+------------------+/       ← b4.show() 표시
/|+----------------+|/
/||*+------------+*||/
/||*|Hello, World.|*||/
/||*+------------+*||/
/|+----------------+|/
/+------------------+/
```

Decorator 패턴

Decorator 패턴