Adapter Design Pattern

תבנית העיצוב Adapter היא תבנית עיצוב מבנית שמאפשרת לאובייקטים עם ממשקים שונים לעבוד ביחד. הAdapter משמש כגשר בין שני ממשקים שונים ומאפשר להם לתקשר זה עם זה.

לתבנית יש שלושה רכיבים עיקריים:

1. **Client** - האובייקט שמשתמש ב-Adapter כדי לתקשר עם ה-Adaptee.
2. **Adaptee** - האובייקט שאנו רוצים להשתמש בו, אך הוא משתמש בממשק שונה מהממשק שה-Client מצפה לו.
3. **Adapter** - האובייקט שמשמש כגשר בין ה-Client ל-Adaptee. ה-Adapter ממיר את הקריאות של ה-Client לקריאות של ה-Adaptee.

יתרונות

- שימוש חוזר - ניתן להשתמש בקוד קיים מבלי לשנות את הממשק.
- גמישות - מנתק את הקשר בין ה-Client ל-Adaptee, ומאפשר להם להיות פחות תלויים זה בזה.
- אם יש שינויים בAdaptee המקום היחיד שיש לשנות הוא ב-Adapter.

דוגמא לשימוש ב- Adapter

נניח שיש לנו תוכנה שמציירת צורות למסך, היא משתמשת בממשק מתקדם בשם `Shape` שבו יש פונקציות בשם `getWidht` , `draw` ו- `getHeight` .

נניח שיש לנו כבר מימוש למלבן אבל עם ממשק ישן יותר שמכיל פונקציות בשם `getY` , `getX` , `diaplay` . כדי להשתמש במלבן הישן עלינו ליצור Adapter שיממיר את הקריאות של ה-Client לקריאות של ה-Adaptee.

אז נגדיר מחלקה חדשה בשם `RectangleAdapter` שמממשת את הממשק `Shape` ומכילה את מופע של `Rectangle` . בפונקציות של ה-Adapter נממיר את הקריאות של המשתמש לקריאות של ה-Adaptee(במקרה שלנו ממירים את `draw` ל- `display` וכו').

```
// Target interface
interface Shape {
    void draw();
    int getWidth();
    int getHeight();
}

// Adaptee (legacy class)
class Rectangle {
    public void display() {
        // ... legacy drawing logic
    }

    public int getX() {
        // ...
    }

    public int getY() {
        // ...
    }
}

// Adapter class
class RectangleAdapter implements Shape {
    private Rectangle rectangle;

    public RectangleAdapter(Rectangle rectangle) {
        this.rectangle = rectangle;
    }

    @Override
    public void draw() {
        rectangle.display();
```

```java
    }

    @Override
    public int getWidth() {
        // Adapt legacy getX() and getY() to calculate width
    }

    @Override
    public int getHeight() {
        // Adapt legacy getX() and getY() to calculate height
    }
}

// Usage
Rectangle legacyRectangle = new Rectangle();
Shape modernShape = new RectangleAdapter(legacyRectangle);

modernShape.draw(); // Uses legacy display() method
```

וככה הצלחנו להתאים את המלבן הישן לממשק החדש ולהשתמש בו בתוכנה שלנו, מבלי לשנות את הממשק הקיים.