Factory Design Pattern

תבנית העיצוב של Factory היא תבנית עיצוב יצירה שמספקת לנו ממשק ליצירת אובייקטים מסוגים שונים ומאפשרת לנו להחליט בזמן ריצה איזה אובייקט ליצור.

התבנית מאפשרת לנו להפריד בין יצירת האובייקט והשימוש בו, וכך להפחית את התלות בין הקוד לבין המחלקה של האובייקט.

יצירת האובייקט היא קופסה שחורה עבור מי שמשתמש באובייקט, והוא לא צריך לדעת איך האובייקט נוצר, זה הופך את הקוד ליותר מודולרי וקל לתחזוקה.

בעצם התבנית זה ליצור מחלקה חדשה, שכל התכלית שלה היא לייצר אובייקטים מסוגים שונים, עם מחלקת אב משותפת.

יתרונות

- אפשר להוסיף בקלות סוגים חדשים של אובייקטים מבלי לשנות את הקוד, אז הוא גם שומר על אינקפסולציה, ועל עקרון ה-Open/Closed.
- מנתק את התלות בין יצירת האובייקט לשימוש בו, ובכך הופך את הקוד ליותר מודולרי וניתן לשימוש חוזר (אם אנחנו יוצרים אובייקטים בכמה מקומות בקוד, לא נצטרך לשנות בכל מקום, רק במחלקת ה-Factory).

דוגמה לשימוש ב- Factory

אנחנו רוצים ליצור תוכנית שמייצרת צורות למסך לפי קלט מהמשתמש שיינתן בזמן ריצת התוכנית.

לצורך כך ניצור ממשק `Shape` ומחלקות מופשטות שמממשות את הממשק, וניצור מחלקה `ShapeFactory` שמייצרת את האובייקטים המתאימים לפי הקלט שניתן לה.

> הפוקנציה `equalsIgnoreCase` משווה בין שתי מחרוזות בלי לשים לב לאותיות גדולות וקטנות, כלומר היא מחשיבה את המחרוזות `CiRcle == circle`.

```java
public interface Shape {
    void draw();
}

public class Circle implements Shape {
    @Override
    public void draw() {
        System.out.println("Drawing a circle");
    }
}

public class Square implements Shape {
    @Override
    public void draw() {
        System.out.println("Drawing a square");
    }
}

// Factory class
public class ShapeFactory {
    public Shape getShape(String shapeType) {
        if (shapeType == null) {
            return null;
        }
        if (shapeType.equalsIgnoreCase("CIRCLE")) {
            return new Circle();
        } else if (shapeType.equalsIgnoreCase("SQUARE")) {
            return new Square();
        }
        return null;
    }
}
```