A GUIDE TO FACTORY PATTERN!

Featuring Cookies!

PROBLEMS

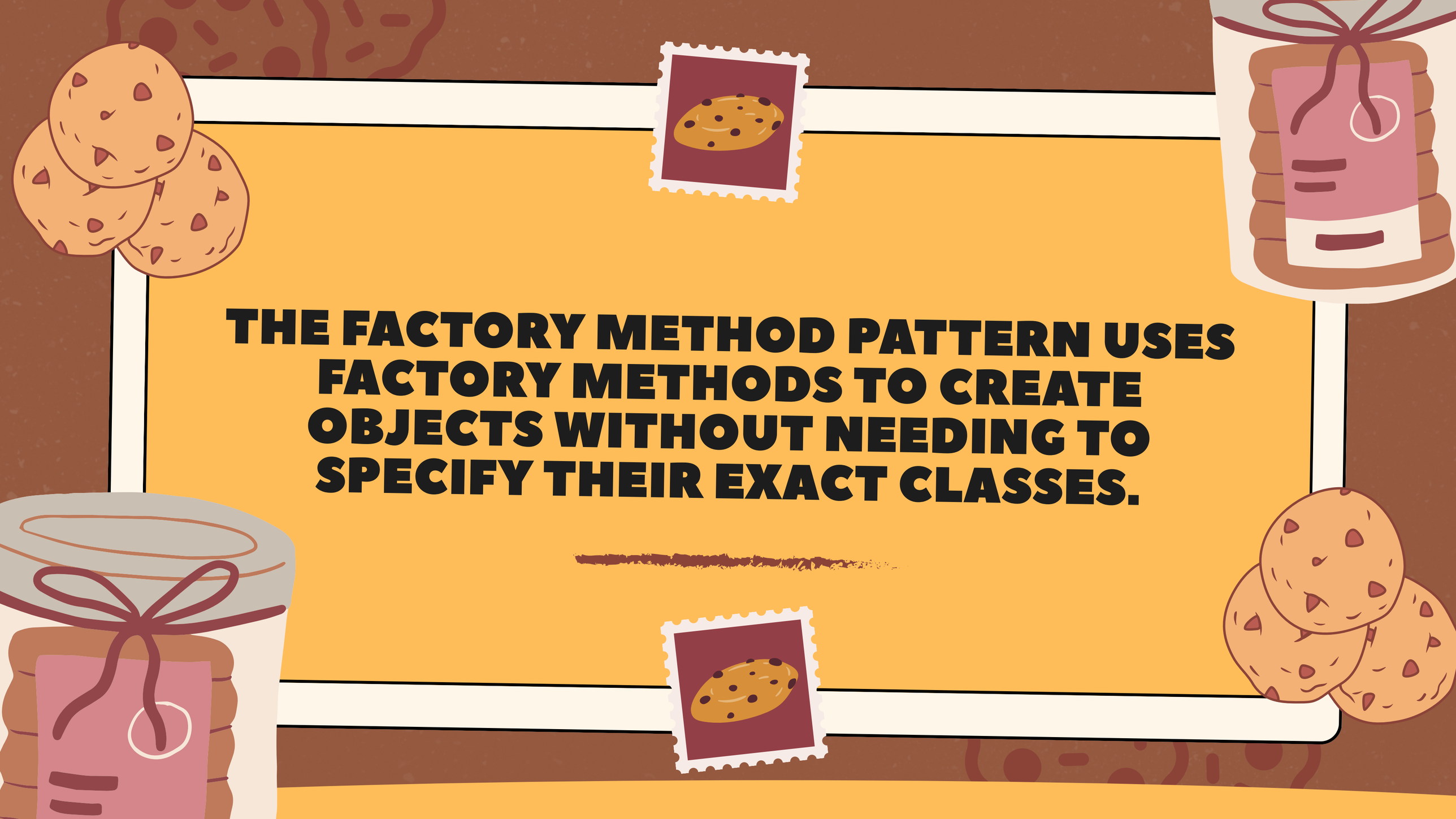
- Imagine you have pre-existing code of a cookie factory.
- However, you only have code for one type of factory that makes chocolate chip cookies.
- What would you do if you needed to make a peanut butter cookie?

PROBLEMS IT SOLVES

- A class needs to create a different type of object but does not know which one to create.
- The Factory Method allows a subclass to create the object. How it creates the object is up to that creator class.
- This pattern centralizes where you create objects to one spot making everything more dynamic and maintainable.

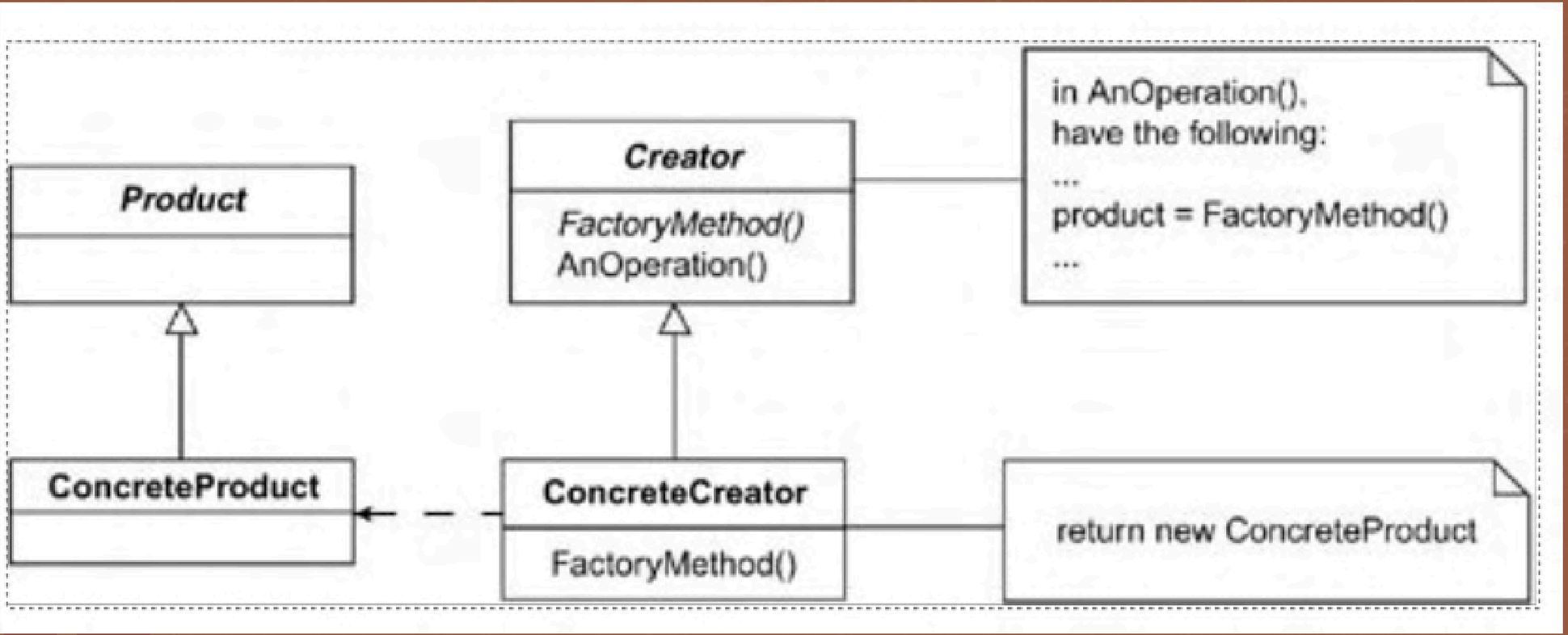
KEY ADVANTAGE

- Encapsulation of object creation
- Logic for object creation is isolated to one location.


THE FACTORY METHOD PATTERN USES FACTORY METHODS TO CREATE OBJECTS WITHOUT NEEDING TO SPECIFY THEIR EXACT CLASSES.

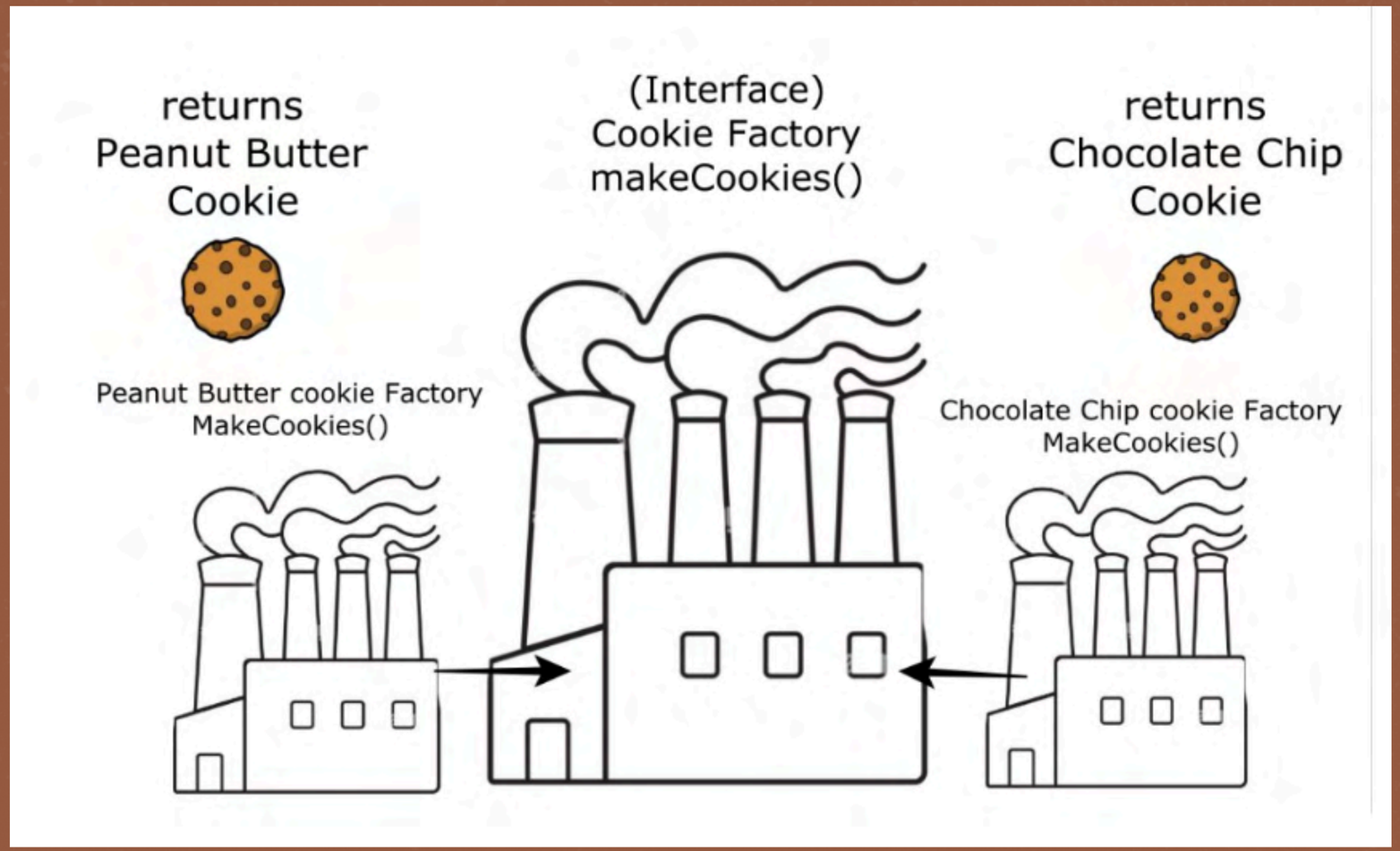
DESIGN

How the pattern is designed!

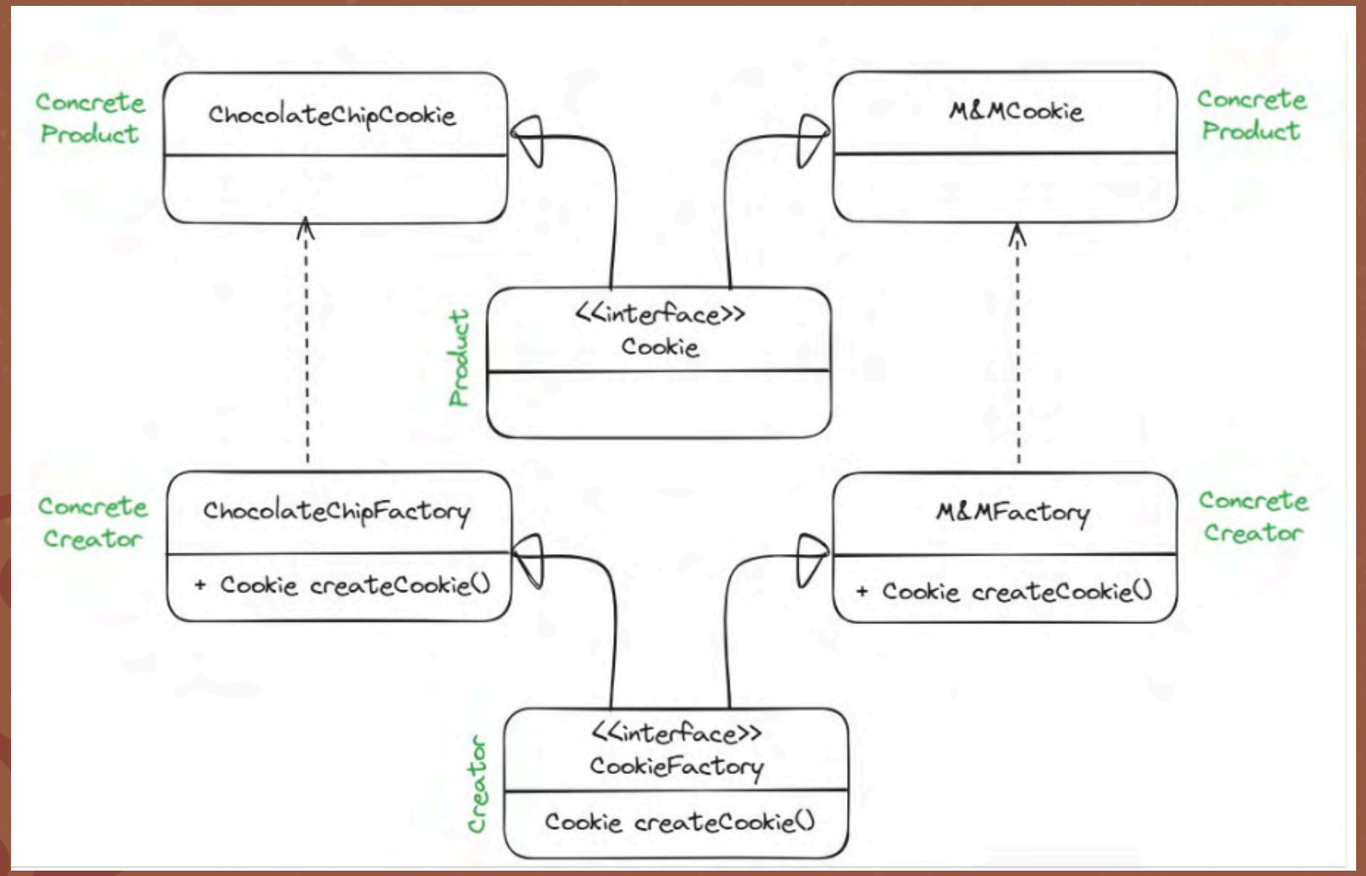


VISUAL

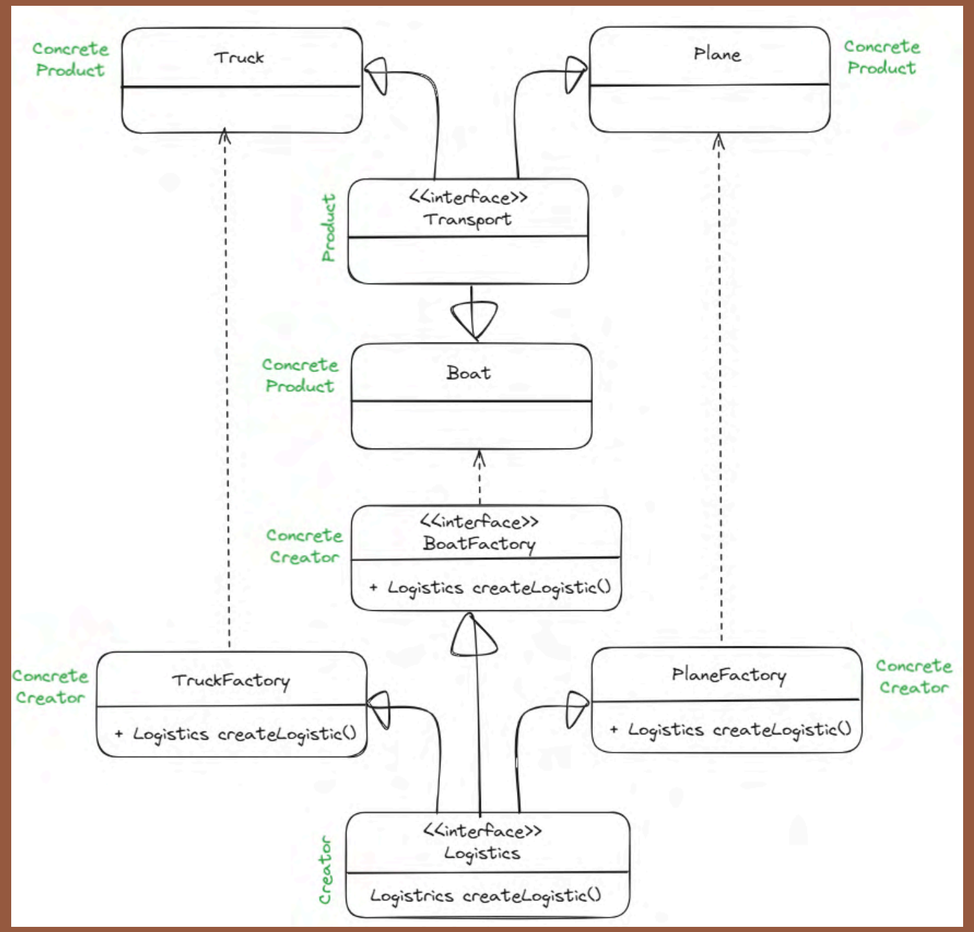
How the pattern is designed!

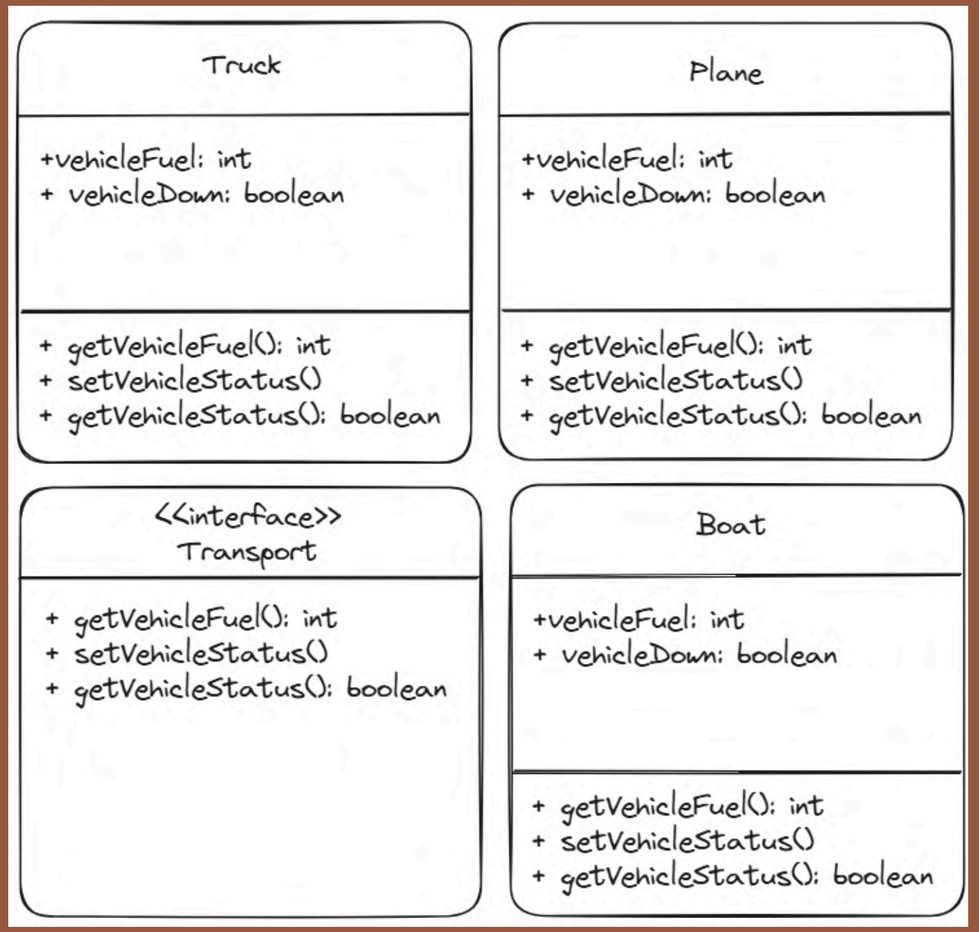


THE COOKIE CLASSES!

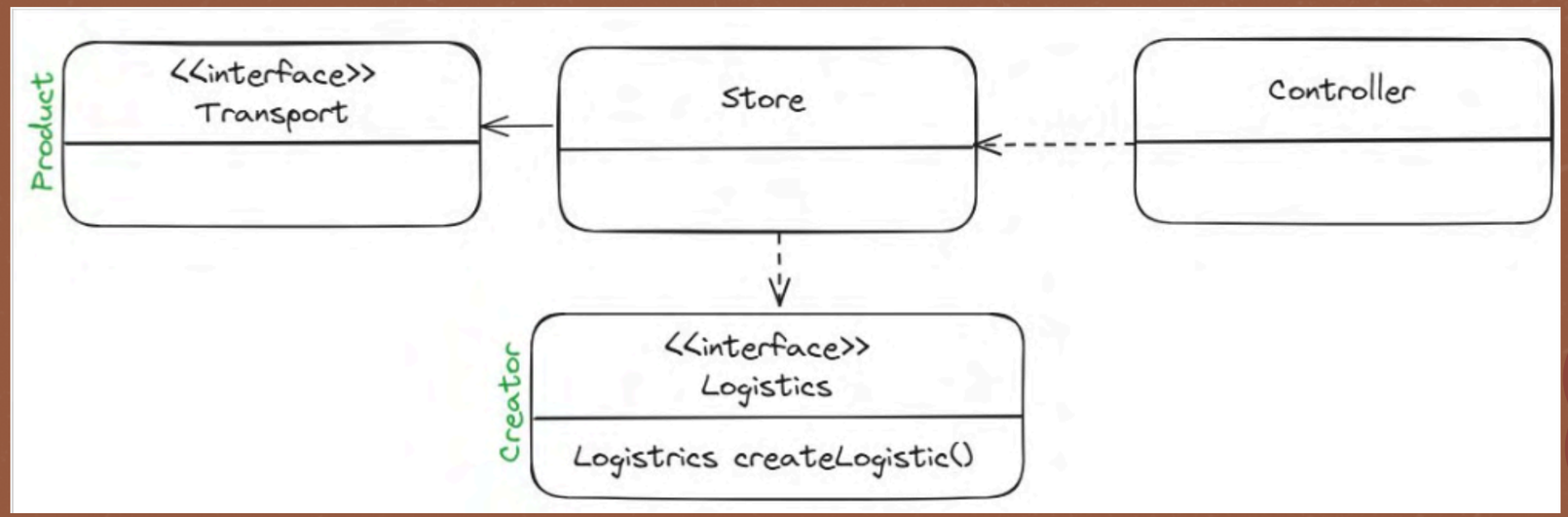


THE TRANSPORT CLASSES!





STORE & CONTROLLER



CODE SNIPPETS!

```java
public class ChocolateChipFactory {

    public Cookie createCookie() {
        return new ChocolateChipCookie(15, 15);
    }

}
```

```java
public interface CookieFactory {

    Cookie createCookie();
}
```

```java
public interface Cookie {

    int getCookiePrice();
    void setCookieBad(boolean cookieBad);
    boolean getCookieStatus();
    void setCookieAge(int cookieAge);
    int getCookieAge();

}
```

```java
public class ChocolateChipCookie implements Cookie {

    private static final int TEN = 10;
    private int cookieBestBy;
    private int cookieAge;
    private boolean cookiesGoneBad;
    private int cookiePrice;


    public ChocolateChipCookie(int cookieAge, int cookiePrice) {
        this.cookieBestBy = TEN;
        this.cookieAge = cookieAge;
        this.cookiesGoneBad = false;
        this.cookiePrice = cookiePrice;
    }
```

```java
@FXML  adamsont +1 *
private void buildFac(){
    if(money >= 1000 && factories.size() < 4 && factoryDropDown.getValue() != null) {
        Pane content = new HBox();
        switch(factoryDropDown.getValue()) {
            case "Chocolate Chip":

                CookieFactory choc = new ChocolateChipFactory();
                factories.add(choc);
                cookieFactories.put(choc, content);
                money -= 1000;
                addTransportationToFactory(choc);
                break;

            case "MnM":

                CookieFactory mnm = new MnMFactory();
                factories.add(mnm);
                cookieFactories.put(mnm, content);
                money -= 1000;
                addTransportationToFactory(mnm);
                break;

            case "Peanut Butter":
                CookieFactory pbf = new PeanutButterFactory();
                factories.add(pbf);
```

RIBS, RIBS & RIBS!

Responsibility

The factory is responsible for object creation

Identity

The type of factory it is and the product it creates.

Behavior

Encapsulates creation logic

State

Minimal with the factory method, but products may have internal states.

PROS & CONS

Pros

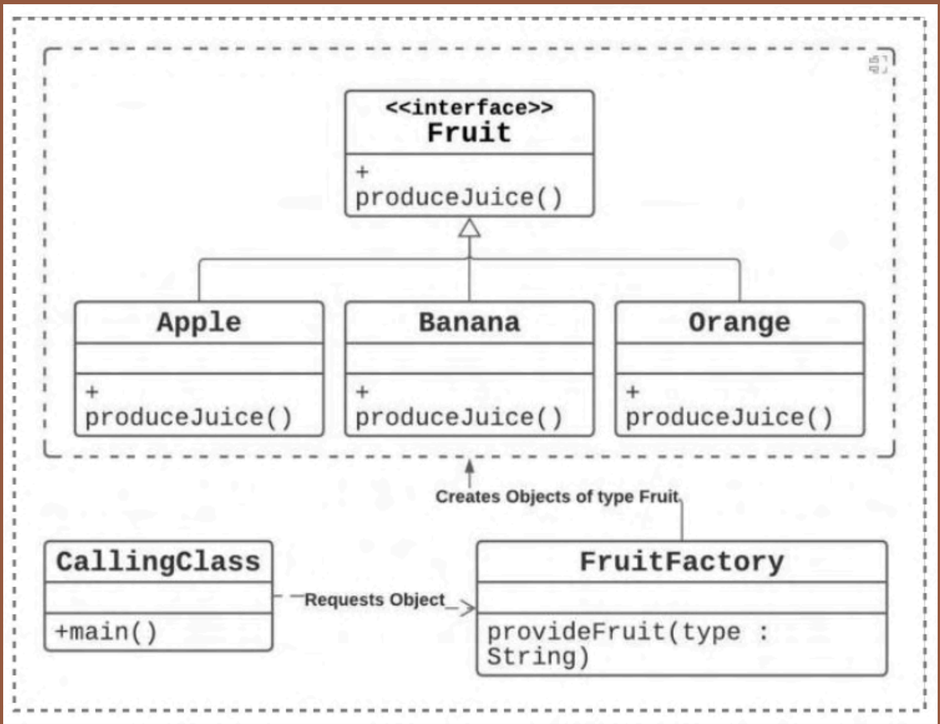
- Low coupling and high cohesion
- Add new elements without breaking previously existing code
- Encapsulates product classes from client.
- Open Closed principle
- Single responsibility principle

Cons

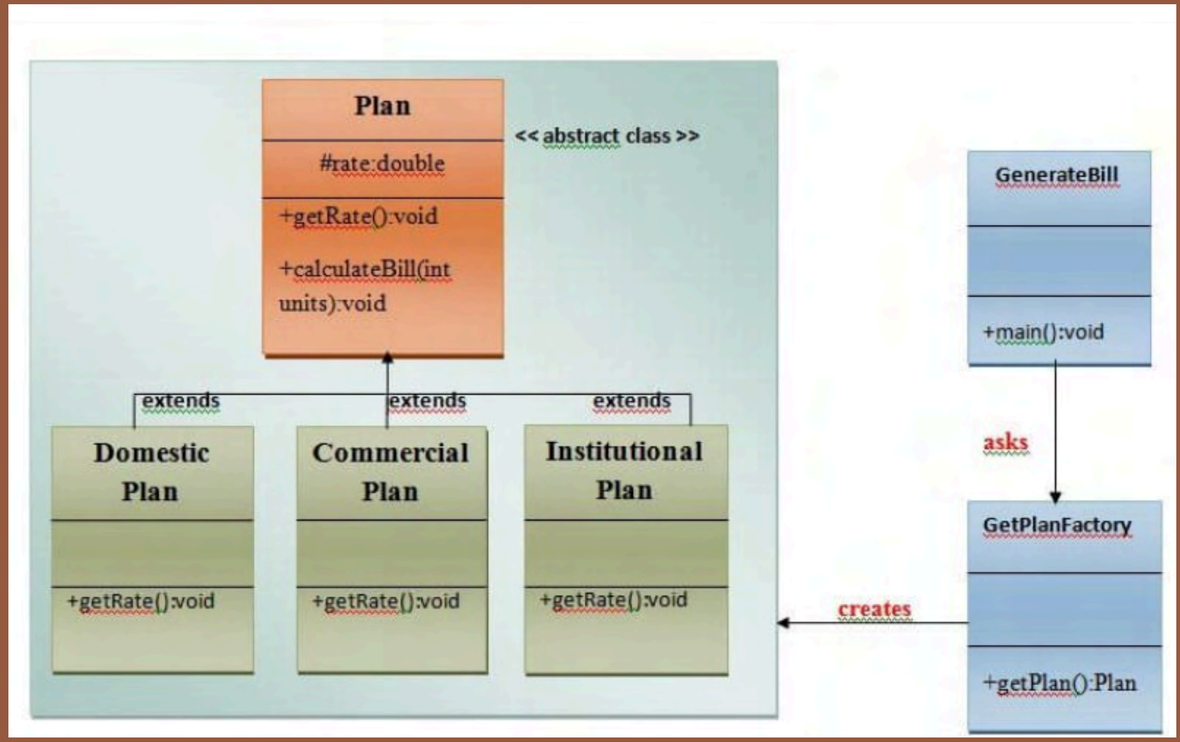
- **Can makes the code complex with a surplus of concrete classes**
- **Client must be aware of concrete subclasses.**

APPLICATIONS!

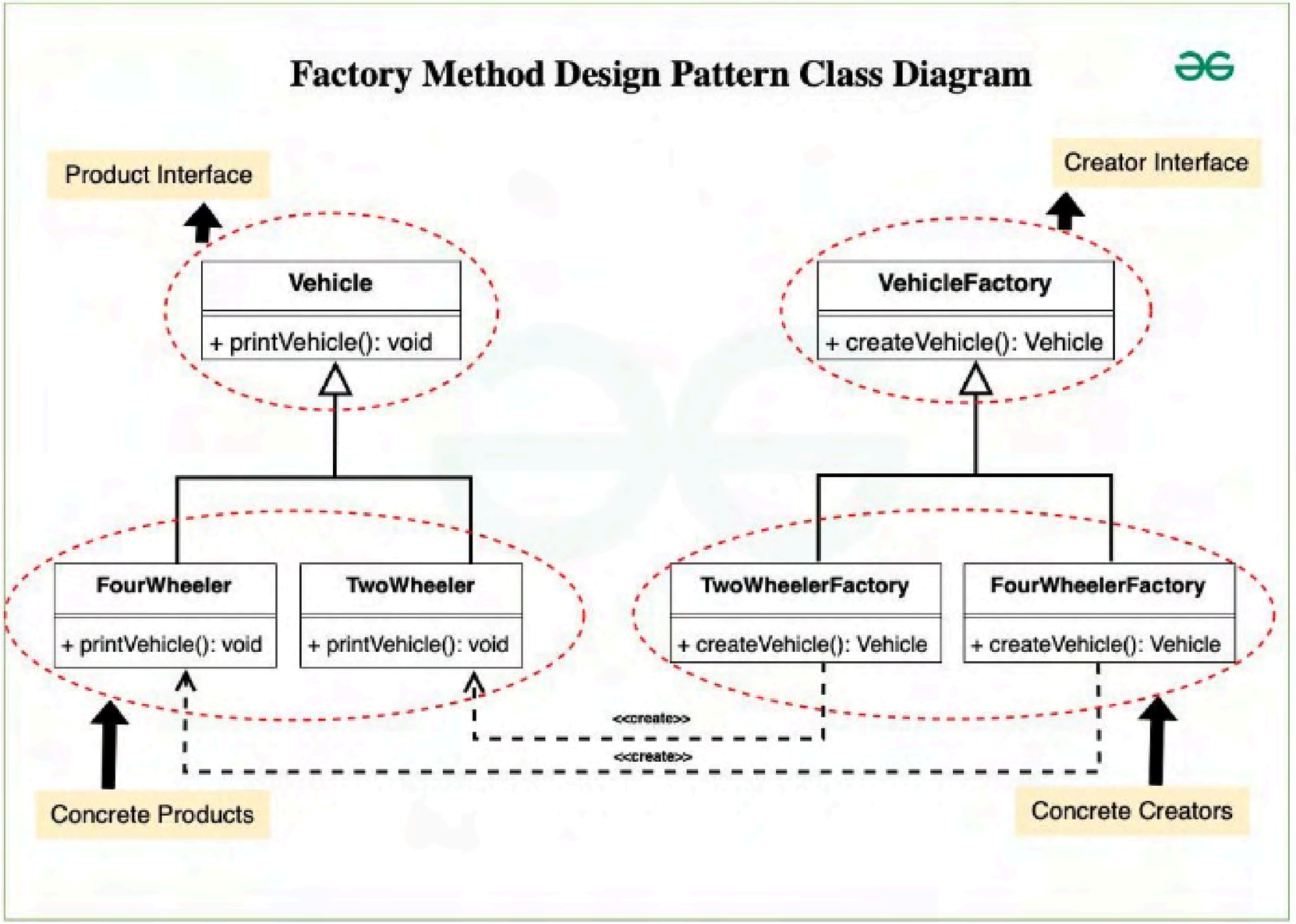
FRUIT FACTORY



BILL FACTORY



VEHICLE FACTORY



CITATIONS

- Gamma, Erich, et al. Design Patterns. Pearson Education, 31 Oct. 1994.
- Refactoring Guru. “Factory Method.” Refactoring.guru, 2014, refactoring.guru/design-patterns/factory-method.
- Anbazhagan, Raja. “Factory Design Pattern.” SpringHow, 29 May 2021, springhow.com/factory-pattern/.
- “Factory Method Design Pattern - Javatpoint.” Www.Javatpoint.Com, www.javatpoint.com/factory-method-design-pattern. Accessed 1 Nov. 2024.
- “Factory Method Pattern.” Wikipedia, Wikimedia Foundation, 30 Oct. 2024, en.wikipedia.org/wiki/Factory_method_pattern.
- GeeksforGeeks, and GeeksforGeeks. “Factory Method Design Pattern.” GeeksforGeeks, 14 Oct. 2024, www.geeksforgeeks.org/factory-method-for-designing-pattern/.
- macrovector. Factory Stock Photo, www.canstockphoto.com/electric-vehicles-production-isometric-79953027.html. Accessed 1 Nov. 2024.