Design Patterns Lec1

Eng/ Mustafa Prince

Requirements

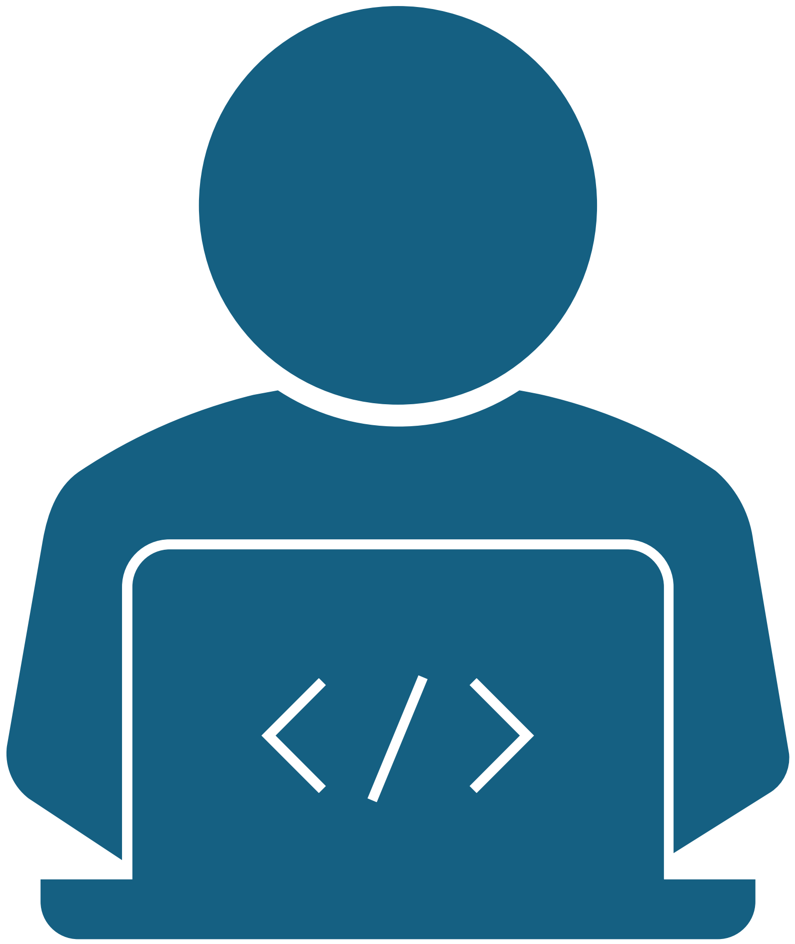
- Any Programming language that support OOP.
- OOP.

Code in this lectures uses dart language.

Sources:

Design Patterns Elements of Reusable Object-Oriented Software book.

Head First Design Patterns book.



Important Concepts

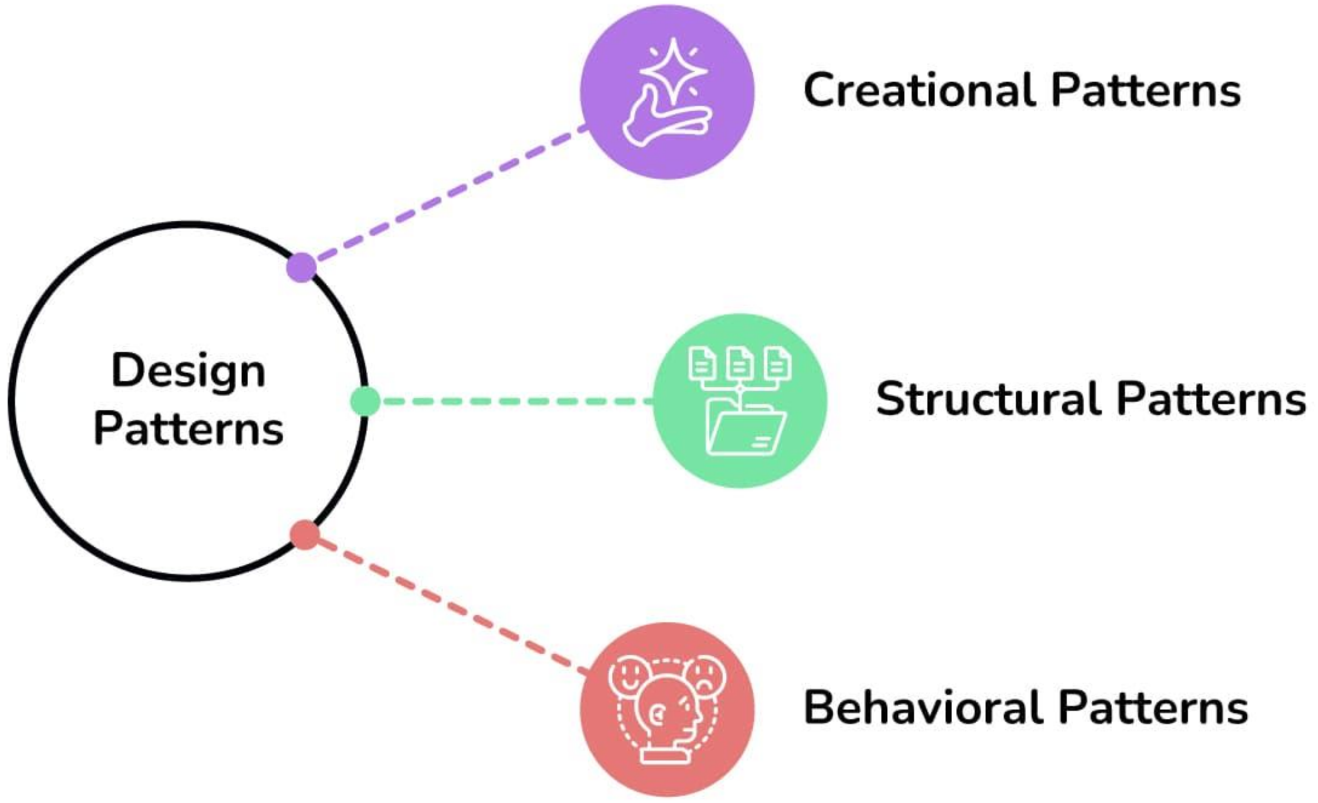
- Architecture Patterns are High-level structures that organize the **overall system design** and responsibilities between layers/modules.
- Solid Principles are Guiding principles (best practices for **code design**).
- Design Patterns are Proven solutions to common problems in **code design.**

Must we apply design patterns?

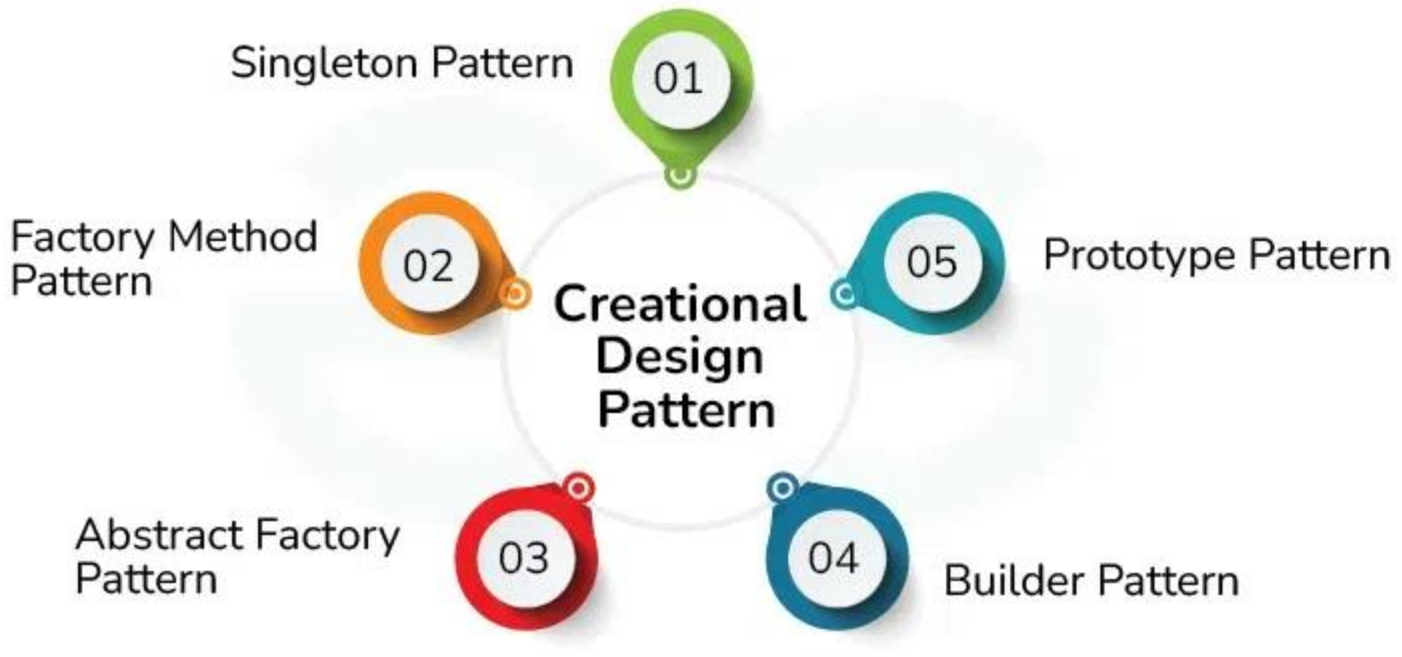
- No, you are not obligated to apply design patterns to every project, but they are highly beneficial for solving common problems, improving code readability, and facilitating communication among developers.

Design Patterns

- There are three types of design patterns.



Creational Design Patterns



Singleton Pattern

- The Singleton Method Design Pattern ensures a class has only one instance and provides a global access point to it.
- In object-oriented programming, the singleton pattern is a software design pattern that restricts the instantiation of a class to a singular instance. It is one of the well-known "Gang of Four" design patterns, which describe how to solve recurring problems in object-oriented software.

Where I can use singleton

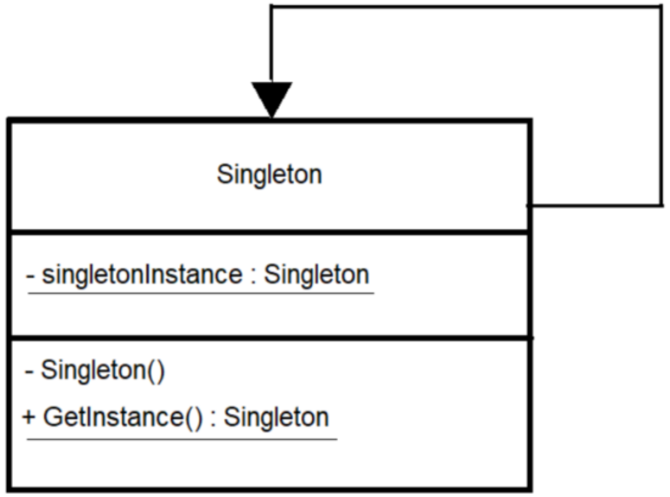
- We need singleton pattern in some applications:
  1. database connection.
  2. logging system.
  3. APIs.

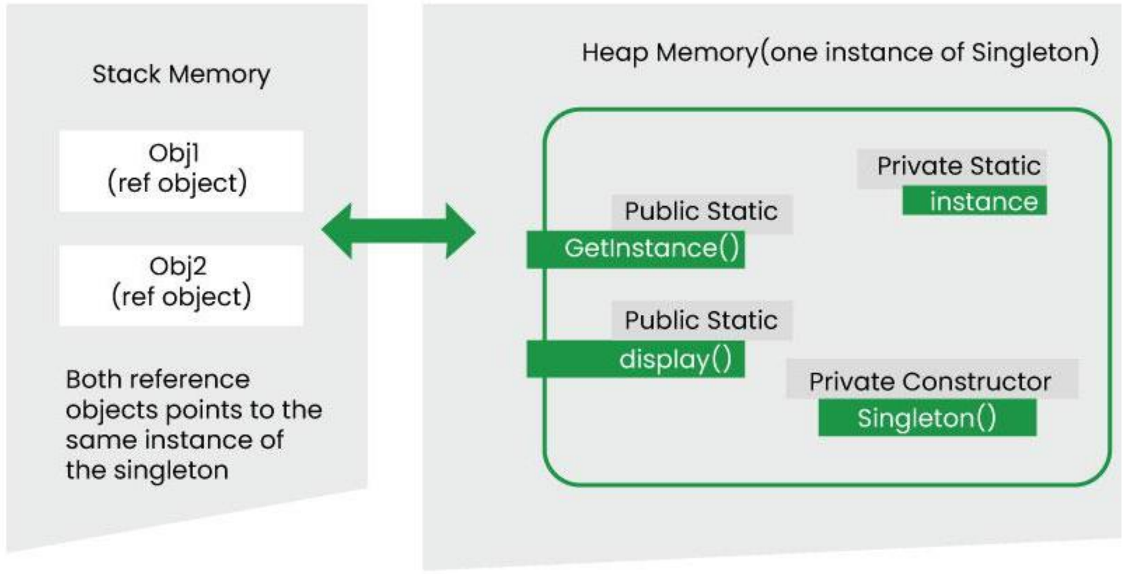
How to apply this pattern?

1. Make the constructor private.
2. Make static private obj from the class in it.
3. Make static public method to return this instance.

This singleton class is now early initialized.

see how to design a lazy initialized singleton.




Stack Memory
Obj1 (ref object)
Obj2 (ref object)
Both reference objects points to the same instance of the singleton
Heap Memory(one instance of Singleton)
Private Static instance
Public Static GetInstance()
Public Static display()
Private Constructor Singleton()

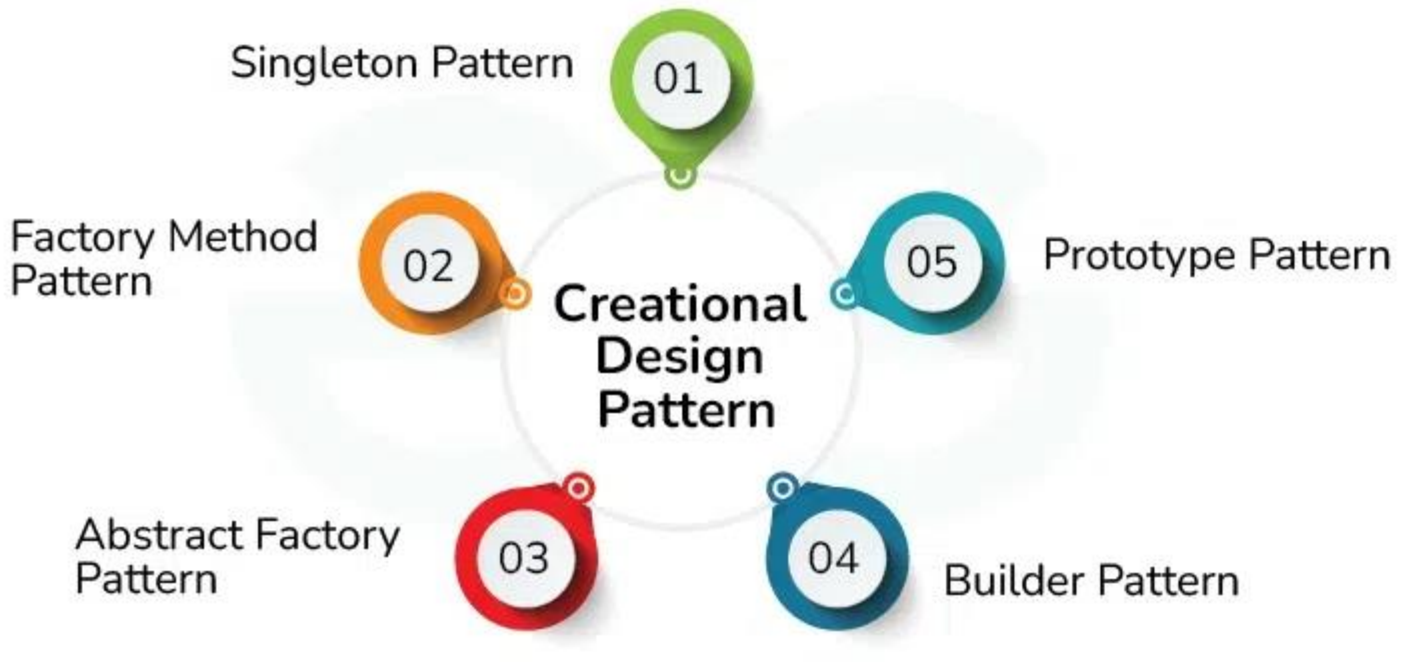
Code Implementation

Thank You

Design Patterns Lec2

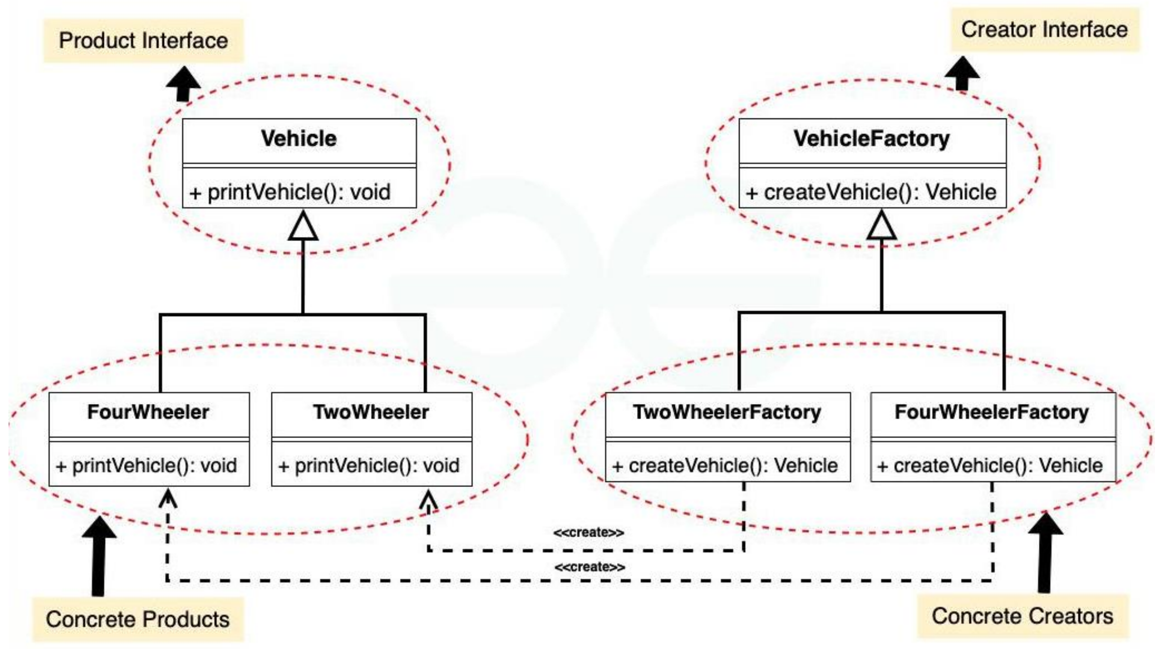
Eng/ Mustafa Prince

Creational Design Patterns



Factory Pattern

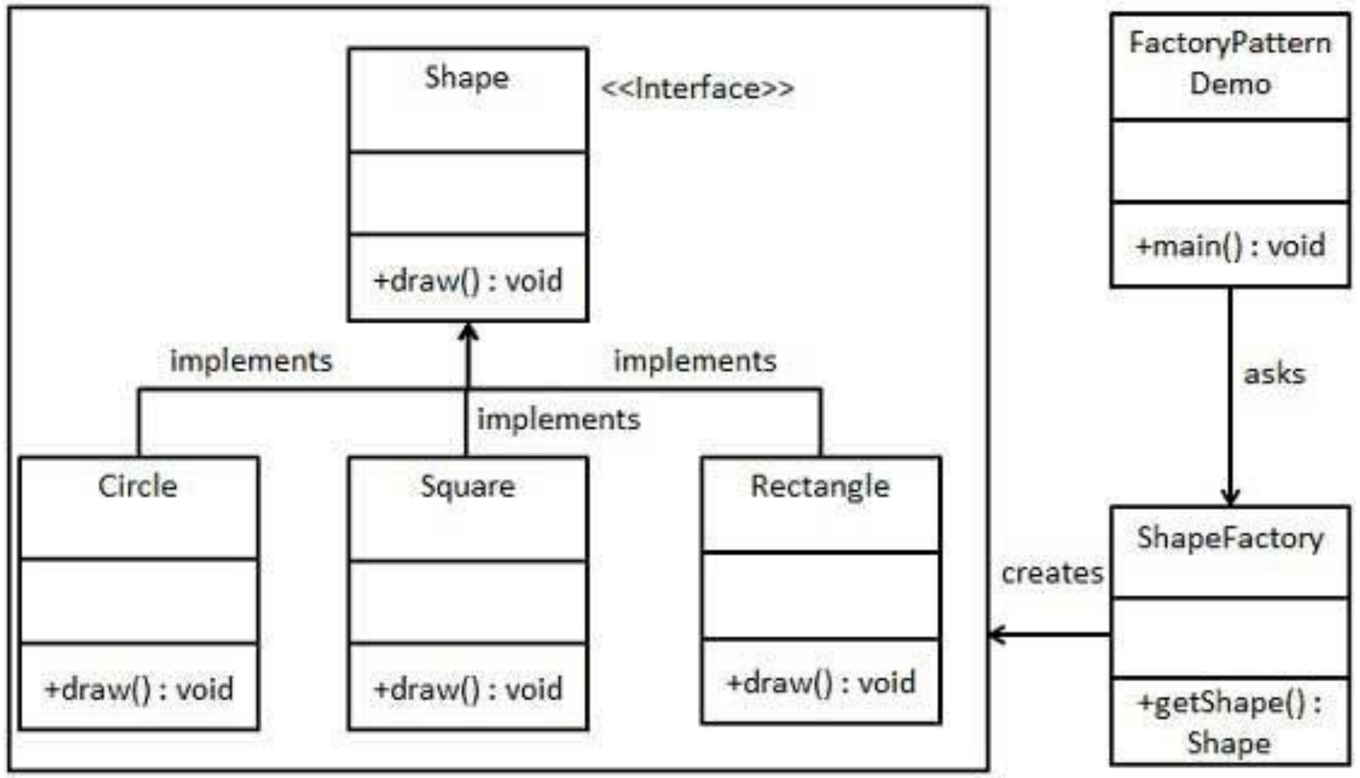
- The Factory Method is a creational design pattern that defines an interface for creating objects in a superclass while allowing subclasses to decide which objects to instantiate.



How to apply this pattern?

1. Create abstract class for product.
2. Create concrete class for each type of the product.
3. Create factory class and decide to return the type of the product according to the input instead of calling it directly in the main.

Components:
1. abstract factory and concrete factory
2. abstract product and concrete products



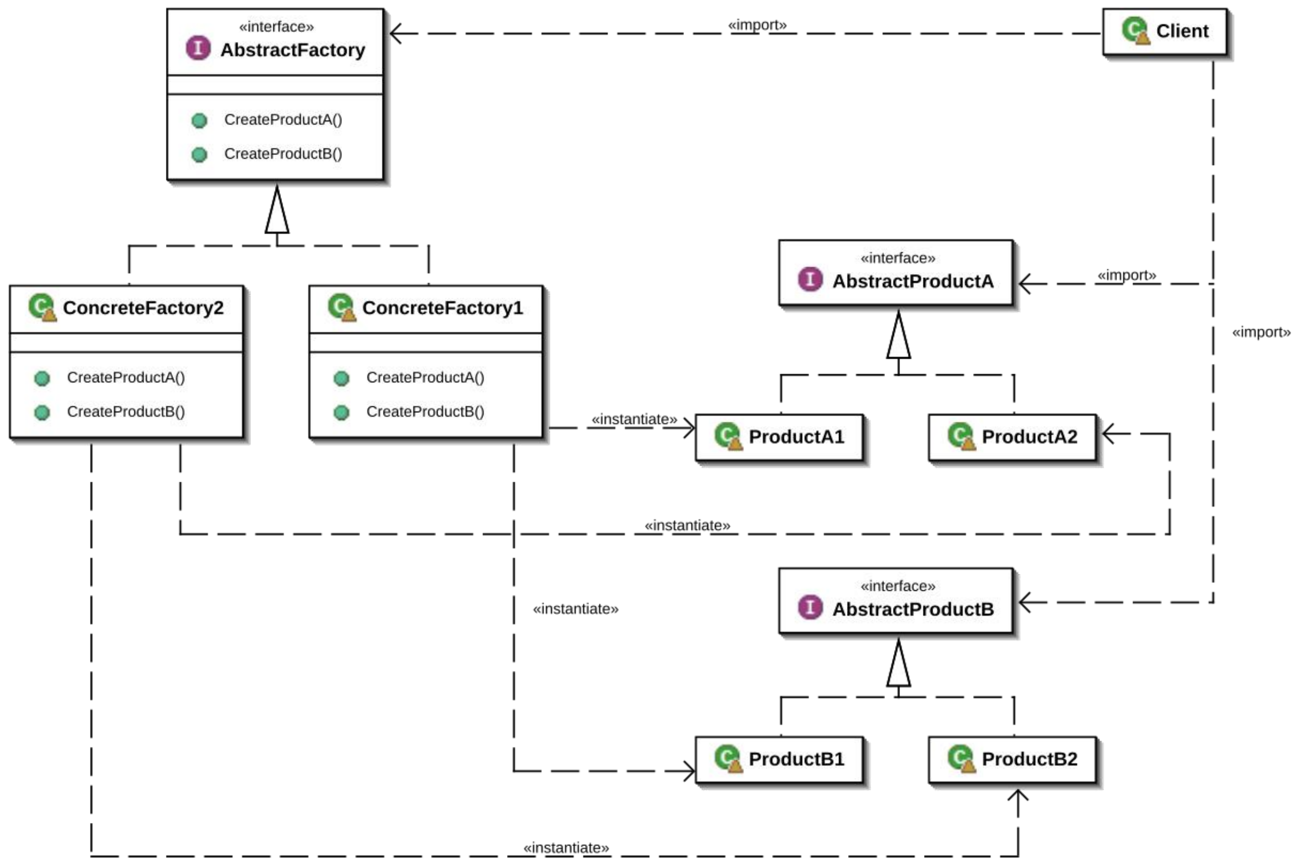
Code Implementation

Where I can use factory pattern

- We need factory pattern in some applications:
  1. Theme switching.
  2. Data source selection.
  3. Navigation.

Abstract Factory Pattern

- The Abstract Factory Pattern is a creational design pattern that provides an interface for creating families of related or dependent objects without specifying their concrete classes.
- Components:
  1. abstract factory + concrete factories
  2. abstract product + concrete products
  3. Client



Code Implementation

Where I can use abstract factory pattern

- We need factory pattern in some applications:
  1. Theme switching.
  2. Localization with languages.
  3. Gamming.

Thank You