# **Creation Patterns**

# **Abstract Factory**

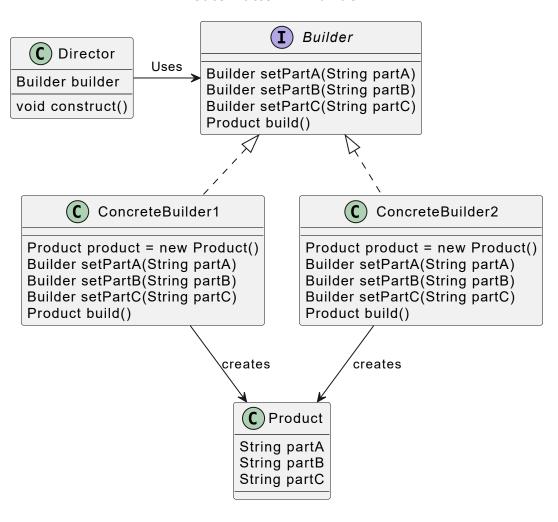
Abstract factories are used to create a set of related objects without specifying their type.

#### Create Pattern - Abstract Factory AbstractFactory AbstractProductA createProductA() AbstractProductB createProductB() C ConcreteFactory1 C ConcreteFactory2 (I) AbstractProductA AbstractProductB AbstractProductA createProductA() AbstractProductA createProductA() void doSomething() void doSomething() AbstractProductB createProductB() AbstractProductB createProductB() creates creates creates creates (C) ProductA1 (C) ProductA2 C ProductB1 C ProductB2 void doSomething() void doSomething() void doSomething() void doSomething()

## Builder

Builder is used to construct an object instance with control flow

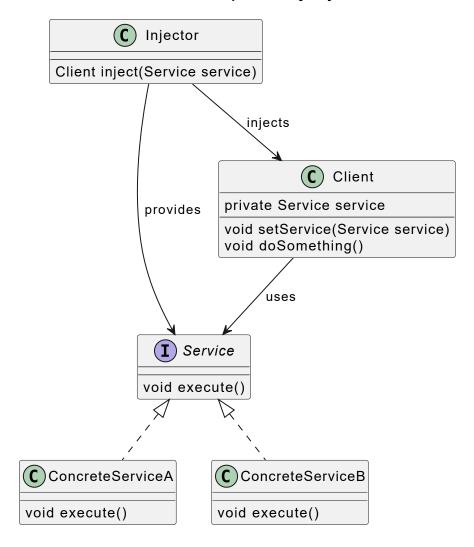
#### Create Pattern - Builder



# **Dependency Injection**

Dependency Injection is a means to control/provide dependent functionality to a class

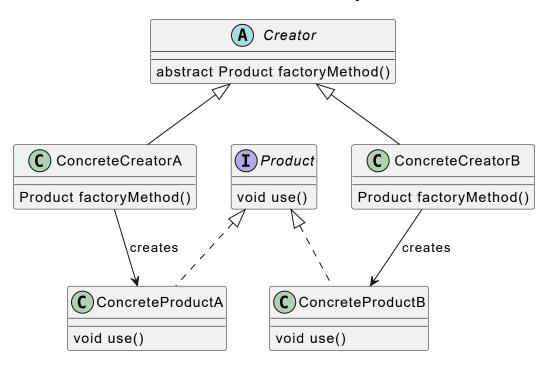
## **Create Pattern - Dependency Injection**



# Factory

Factories are used to create the particular type of class instance

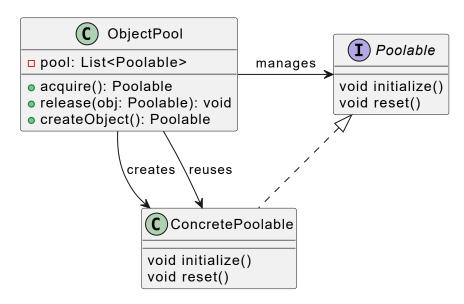
## **Create Pattern - Factory**



## **Pool**

Object pools provide a means to reuse object instances that are expensive to create. Examples may include data service connection or threads.

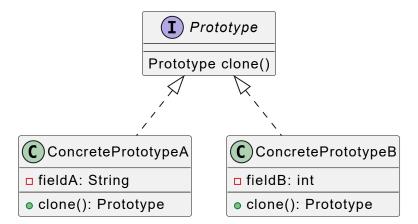
#### Create Pattern - Pool



## Prototype

Prototypes create instances by copying an existing object

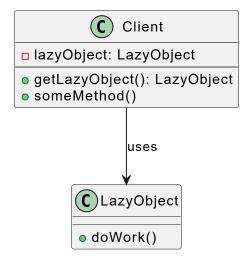
## Create Pattern - Prototype



## Lazy Initialization

Proxies or Lazy Initialized objects may be used for objects that are expensive to create. The actual creation of the object is not called until the first time the object is required.

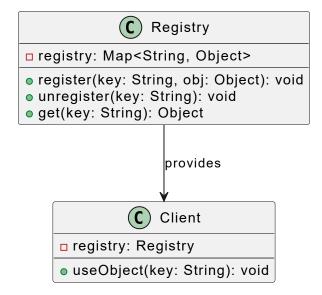
### Create Pattern - Lazy Initialization



# Registry (Multiton)

A registry allowed for storing/re-using a collect of named instances of an object.

## Create Pattern - Registry



# Singleton

Singletons are used to ensure only a single instance of an object is created in the lifetime of the application.

## Create Pattern - Singleton

