SWIFT
Factory Method

Bill Kim(김정훈) | ibillkim@gmail.com

목차

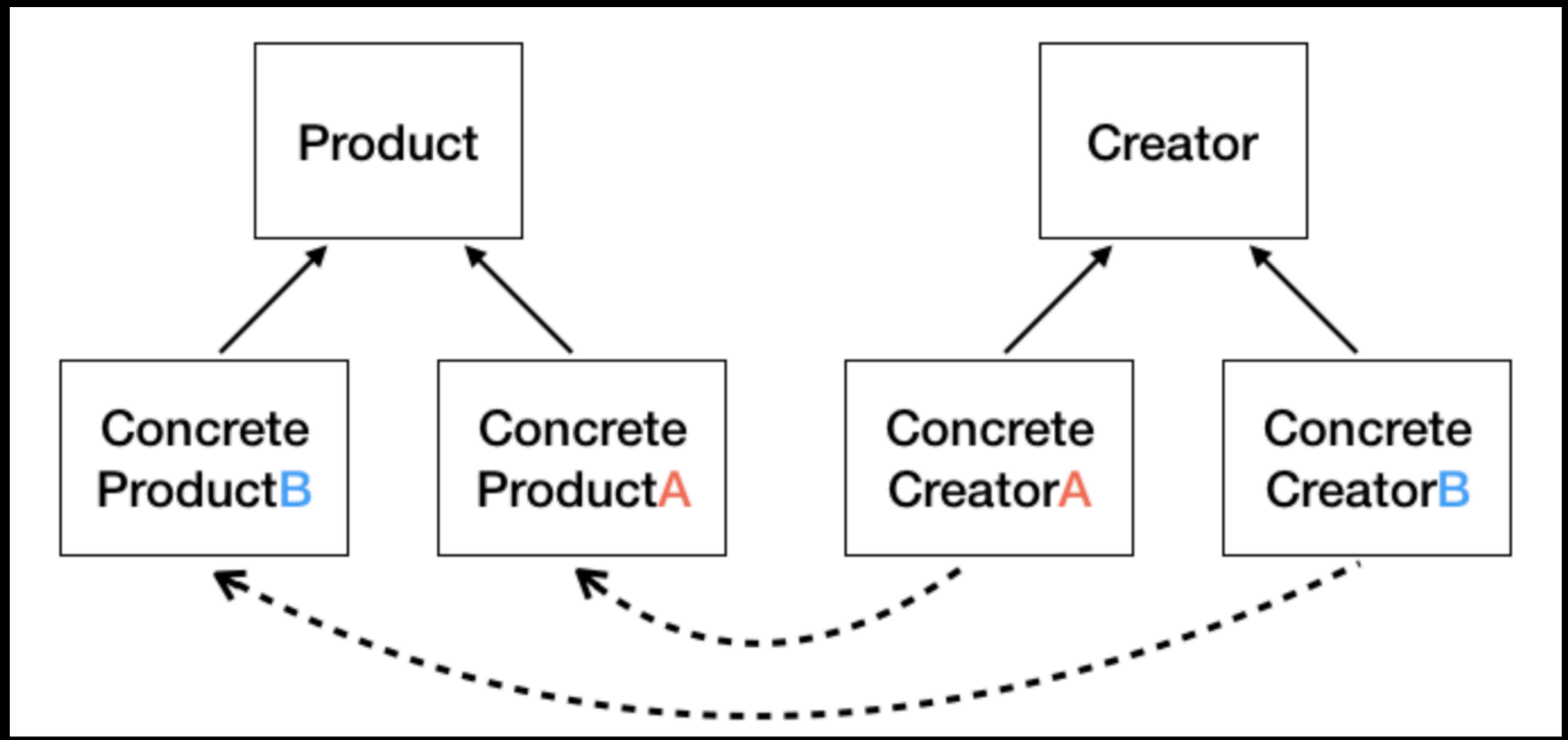
Factory Method

Factory Mthod(팩토리 매소드) 디자인 패턴은 객체 생성을 위해 인터페이스는 정의하지만 어떤 클래스의 인스턴스를 생성할 지에 결정은 서브클래스가 정의하도록 해주는 디자인 패턴입니다.

팩토리 메서드는 직접 인스턴스를 생성하는 대신 생성을 위한 메서드를 인터페이스로 제공합니다. 서브 클래스는 생성될 객체의 클래스를 변경하기 위해서 메서드를 재정의 할 수 있습니다.

Factory Method

팩토리 매소드 패턴을 위해서 필요한 객체 구성은 아래와 같습니다.



Structure

Creator : 추상 팩토리 메소드 정의 및 팩토리 메소드를 호출하여 Product를 생성하기 위한 객체

Product : 팩토리 메소드에 의해 성성되는 기본 인터페이스 객체

Concrete Creator : 팩토리 메소드를 구현하고 Concrete Product 인스턴스를 반환하는 객체

Concrete Product : Product를 구현하는 객체

Implementation

구체적인 구현에 대해서 소스 코드를 통하여 살펴봅니다.

```
// Creator
protocol Creator {
    func factoryMethod() -> Product
    func someOperation() -> String
}

extension Creator {
    func someOperation() -> String {
        let product = factoryMethod()
          return product.operation()
    }
}

// Product
protocol Product {
    func operation() -> String
}
```

Implementation

```
// Concrete Creators
class ConcreteCreator1: Creator {
    public func factoryMethod() -> Product {
        return ConcreteProduct1()
    }
}

class ConcreteCreator2: Creator {
    public func factoryMethod() -> Product {
        return ConcreteProduct2()
    }
}

// Concrete Products
class ConcreteProduct1: Product {
    func operation() -> String {
        print("ConcreteProduct1 operation")
        return "ConcreteProduct1 operation"
    }
}

class ConcreteProduct2: Product {
    func operation() -> String {
        print("ConcreteProduct2 operation")
        return "ConcreteProduct2 operation"
    }
}
```

Implementation

```
func concreteClient(creator: Creator) {
    creator.someOperation()
}

concreteClient(creator: ConcreteCreator1())
// ConcreteProduct1 operation

concreteClient(creator: ConcreteCreator2())
// ConcreteProduct2 operation
```

References

[1] 팩토리 메서드 패턴 : https://jerome.kr/entry/factory-method-pattern?category=1114713

[2] [Swift] 팩토리 메서드(Factory Method) 패턴 : https://m.blog.naver.com/PostView.nhn?blogId=horajjan&logNo=220804516206&categoryNo=97&proxyReferer=&proxyReferer=https:%2F%2Fwww.google.com%2F

[3] Design Pattern - Factory : https://ehdrjsdlzzzz.github.io/2019/04/03/Design-Pattern-Factory/

[4] Swift Solutions: Factory Method Pattern : https://swiftcraft.io/swift-solutions-factory-method-pattern/

[5] Factory Method in Swift : https://refactoring.guru/design-patterns/factory-method/swift/example

References

[6] Swift factory method design pattern : https://theswiftdev.com/swift-factory-method-design-pattern/

[7] Factory Method in Swift : https://medium.com/jeremy-codes/factory-method-in-swift-d5222dd6e61d

[8] [Design Pattern] Factory method pattern : https://medium.com/@eyegochild/design-pattern-factory-method-pattern-dc72f35e1076

[9] 디자인 패턴 : 추상 팩토리 vs 팩토리 메소드 : https://www.it-swarm.dev/ko/design-patterns/디자인-패턴-추상-팩토리-vs-팩토리-메소드/970371311/

[10] Design Patterns in Swift #1 Factory Method and Singleton : https://www.appcoda.com/design-pattern-creational/

Thank you!