SWIFT
Prototype

Bill Kim(김정훈) | ibillkim@gmail.com

목차

Prototype

프로토타입(Prototype) 디자인 패턴은 객체 생성을 위한 패턴으로서 클래스 정의와 생성 방식을 구조화 및 캡슐화하여 수행할 수 있도록 도와주는 디자인 패턴입니다.

프로토타입 패턴은 원형이 되는 인스턴스를 사용하여 생성할 객체의 종류를 명시하고, 이렇게 만든 견본을 복사해서 새롭게 객체를 생성하여 사용하는 패턴입니다.

구체 클래스를 알 수 없는 경우에도 객체를 복사할 수 있는 공통된 인터페이스를 제공합니다.

동일한 클래스의 객체는 내부 프로퍼티를 모두 알 수 있으므로 프로토타입 객체는 전체 복사본을 생성할 수 있습니다.

Prototype

프로토타입의 주요 구조를 요약하면 아래와 같습니다.

1. 공통의 인터페이스를 갖는 Prototype 객체를 선언합니다.

2. Prototype 객체의 clone 함수를 통하여 새로운 ConcreteProtype 객체를 생성하여 사용할 수 있습니다.

3. 새로운 ConcreteProtype 객체가 추가되더라도 Client 객체는 수정될 필요가 없습니다.

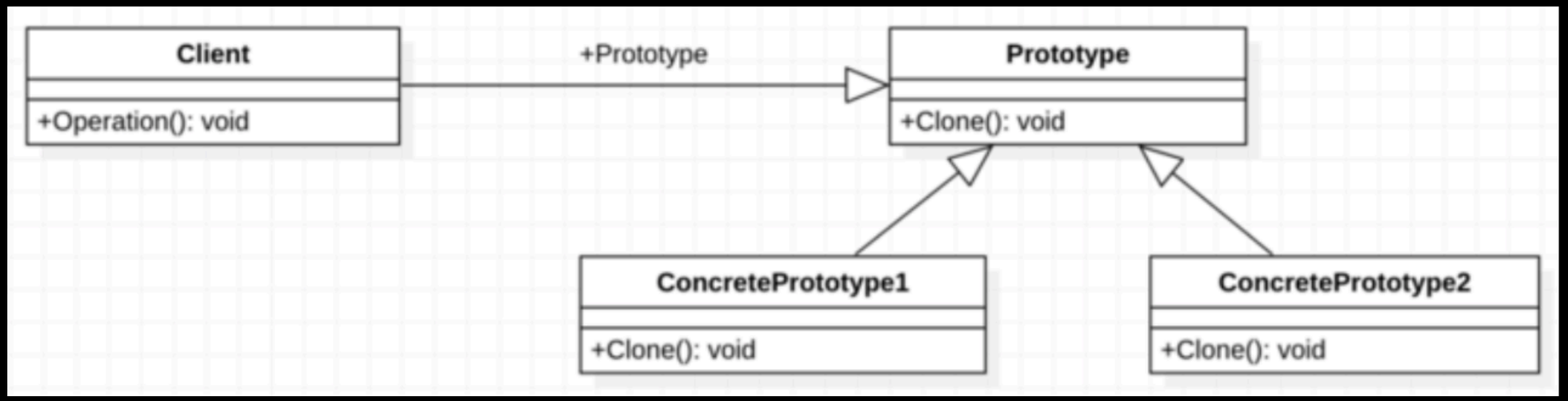
Structure

크게 구조를 보면 아래와 같은 구조를 갑습니다.

Prototype : 객체 복제(Clone)를 위한 원형 인터페이스 객체

ConcretePrototype : 자신을 복제하는 객체

Client : 실제적으로 복제를 요청하고 사용할 객체



Implementation

구체적인 구현에 대해서 소스 코드를 통하여 살펴봅니다.

```
protocol Prototype {
    func operation(value: Int)
    func clone() -> Prototype
}

class ConcretePrototypeA : Prototype {
    init() { }
    init(client: ConcretePrototypeA) { }

    func operation(value: Int) { print("ConcretePrototypeA operation - value : \(value)") }

    func clone() -> Prototype {
        print("ConcretePrototypeA clone")
        return ConcretePrototypeA(client: self)
    }
}

class ConcretePrototypeB : Prototype {
    init() { }
    init(client: ConcretePrototypeB) { }

    func operation(value: Int) { print("ConcretePrototypeB operation - value : \(value*2)") }

    func clone() -> Prototype {
        print("ConcretePrototypeB clone")
        return ConcretePrototypeB(client: self)
    }
}
```

Implementation

```
func concretePrototype(prototype: Prototype) -> Prototype
{
    // Client는 구체적인 ConcretePrototype 객체 접근 불필요
    // 모든 ConcretePrototype 객체에 대해 알 필요가 없다.
    // 새로운 ConcretePrototype 객체가 추가되더라도 객체 생성 시 수정이 불필요

    let newPrototype = prototype.clone()
    return newPrototype
}
```

```
let clientA = concretePrototype(prototype: ConcretePrototypeA())
clientA.operation(value: 10)
// ConcretePrototypeA clone
// ConcretePrototypeA operation - value : 10

let clientB = concretePrototype(prototype: ConcretePrototypeB())
clientB.operation(value: 10)
// ConcretePrototypeB clone
// ConcretePrototypeB operation - value : 20
```

References

[1] Prototype pattern in Swift : https://medium.com/jeremy-codes/prototype-pattern-in-swift-1b50517d1075

[2] 프로토타입 패턴 (Prototype Pattern in Swift) : https://jerome.kr/entry/prototype-pattern?category=1114713

[3] Swift prototype design pattern : https://theswiftdev.com/swift-prototype-design-pattern/

[4] Prototype in Swift : https://refactoring.guru/design-patterns/prototype/swift/example

[5] PatternPrototype : https://github.com/qbbang/PatternPrototype

References

[6] [소프트웨어 공학 - Prototype Pattern] : https://m.blog.naver.com/scw0531/221465259625

[7] [Design Pattern] 프로토타입(Prototype) 패턴 - 디자인 패턴 : https://palpit.tistory.com/189

[8] DesignPattern : 프로토타입 (ProtoType) : http://egloos.zum.com/EireneHue/v/976506

[9] 프로토타입 패턴 : https://ko.wikipedia.org/wiki/프로토타입_패턴

[10] Design Patterns in Swift: Prototype Pattern : https://swiftscheming.wordpress.com/2016/04/17/design-patterns-in-swift-prototype-pattern/

Thank you!