SWIFT
Memento

Bill Kim(김정훈) | ibillkim@gmail.com

목차

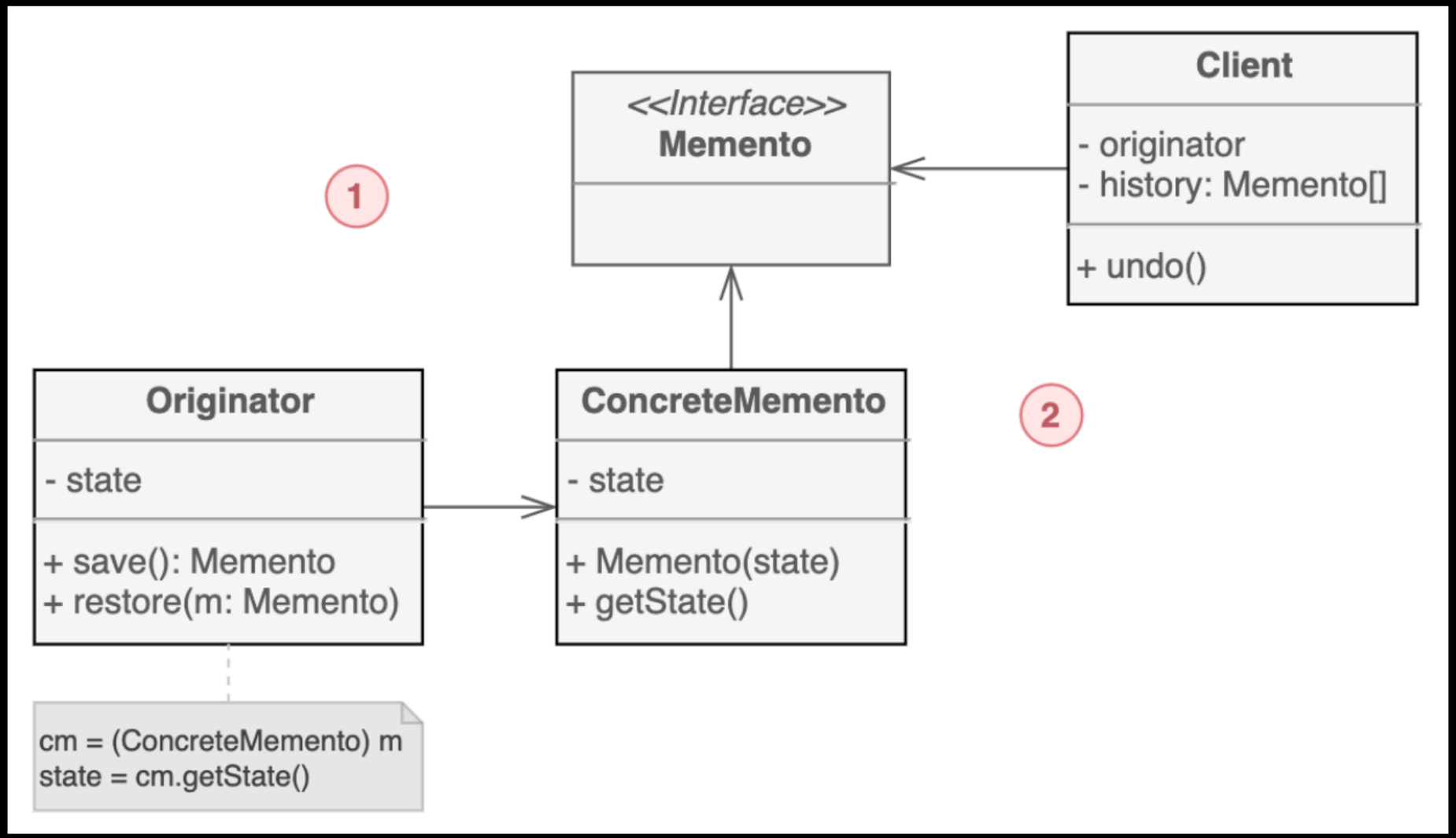
Memento

Memento(메멘토) 패턴은 객체의 상태를 저장하고 복원할 수 있는 행동 디자인 패턴입니다.

기본 오리지날 객체는 본래의 기능에만 충실하고 상태 정보에 대한 관리를 메멘토 객체에게 전담하여 상태를 저장하고 복원할 수 있게 한다.

Structure

Memento 패턴을 UML로 도식화하면 아래와 같습니다.



Structure

Originator : 상태값을 가지고 있고 객체의 본연의 기능만을 수행하며 저장 및 복원에 대한 작업은 Memento 객체에게 넘기는 객체

Memento : Originator 객체의 스냅샷 역할을 하는 값 객체이다. 메멘토 데이터는 생성장를 통해서 한번만 전달하는 것이 일반적입니다.

ConcreteMemento : Memento 객체를 상속받아 메멘토 객체가 가진 데이터에 대한 초기화를 진행하는 클래스 객체

Caretaker(Client) : Originator 객체에 대한 상태가 언제 바뀌고 저장되었는지를 알고 있는 객체, 또한 스냅샷 리스트를 가지고 있고 복원을 할 경우 가장 최상위의 Originator 상태로 복원을 진행하는 객체

Implementation

구체적인 구현에 대해서 소스 코드를 통하여 살펴봅니다.

```
class Originator {
    private var state: String

    init(state: String) {
        self.state = state
        print("Originator: My initial state is: \(state)")
    }

    func doSomething() {
        print("Originator: I'm doing something important.")
        state = generateRandomString()
        print("Originator: and my state has changed to: \(state)")
    }

    private func generateRandomString() -> String {
        return String(UUID().uuidString.suffix(4))
    }

    func save() -> Memento {
        return ConcreteMemento(state: state)
    }

    func restore(memento: Memento) {
        guard let memento = memento as? ConcreteMemento else { return }
        self.state = memento.state
        print("Originator: My state has changed to: \(state)")
    }
}
```

Implementation

```
protocol Memento {
    var name: String { get }
    var date: Date { get }
}

class ConcreteMemento: Memento {
    private(set) var state: String
    private(set) var date: Date

    init(state: String) {
        self.state = state
        self.date = Date()
    }

    var name:String { return state + " " + date.description.suffix(14).prefix(8) }
}
```

Implementation

```
class Caretaker {
    private lazy var mementos = [Memento]()
    private var originator: Originator

    init(originator: Originator) {
        self.originator = originator
    }

    func backup() {
        print("\nSaving Originator's state...\n")
        mementos.append(originator.save())
    }

    func undo() {
        guard !mementos.isEmpty else { return }
        let removedMemento = mementos.removeLast()

        print("Restoring state to: " + removedMemento.name)
        originator.restore(memento: removedMemento)
    }

    func showHistory() {
        print("Here's the list of mementos:\n")
        mementos.forEach({ print($0.name) })
    }
}
```

Implementation

```
let originator = Originator(state: "Super-duper-super-puper-super.")
// Originator: My initial state is: Super-duper-super-puper-super.

let caretaker = Caretaker(originator: originator)

caretaker.backup()
// Saving Originator's state...

originator.doSomething()
// Originator: I'm doing something important.
// Originator: and my state has changed to: B147

caretaker.backup()
// Saving Originator's state...

originator.doSomething()
// Originator: I'm doing something important.
// Originator: and my state has changed to: 20B2

caretaker.backup()
// Saving Originator's state...
```

Implementation

```
caretaker.showHistory()
// Super-duper-super-puper-super. 08:22:02
// B147 08:22:02
// 20B2 08:22:02

caretaker.undo()
// Restoring state to: 20B2 08:22:31
// Originator: My state has changed to: 20B2

caretaker.undo()
// Restoring state to: B147 08:22:31
// Originator: My state has changed to: B147

caretaker.undo()
// Caretaker: Restoring state to: Super-duper-super-puper-super. 08:22:31
// Originator: My state has changed to: Super-duper-super-puper-super.
```

References

[1] Memento in Swift : https://refactoring.guru/design-patterns/memento/swift/example

[2] 메멘토 패턴 (Memento Pattern in Swift) : https://jerome.kr/entry/memento-pattern

[3] Design Patterns in Swift #2: Observer and Memento : https://www.appcoda.com/design-pattern-behavorial/

[4] A Design Pattern Story in Swift - Chapter 17: Memento : http://audreyli.me/2015/07/15/a-design-pattern-story-in-swift-chapter-17-memento/

[5] How to Use the Memento Pattern : https://aruniphoneapplication.blogspot.com/2016/12/the-memento-pattern.html

References

[6] [Design Pattern] 메멘토(Memento) 패턴 - 디자인 패턴 : https://palpit.tistory.com/205

[7] Design Patterns - Memento Pattern : https://www.tutorialspoint.com/design_pattern/memento_pattern.htm

[8] Design Pattern: Memento Pattern : https://viblo.asia/p/design-pattern-memento-pattern-eW65GDPOKDO

[9] 메멘토 패턴 : https://ko.wikipedia.org/wiki/메멘토_패턴

[10] Memento Design Pattern : https://sourcemaking.com/design_patterns/memento

Thank you!