객체설계
현실과 똑같을 필요는 없다

Drake

순서

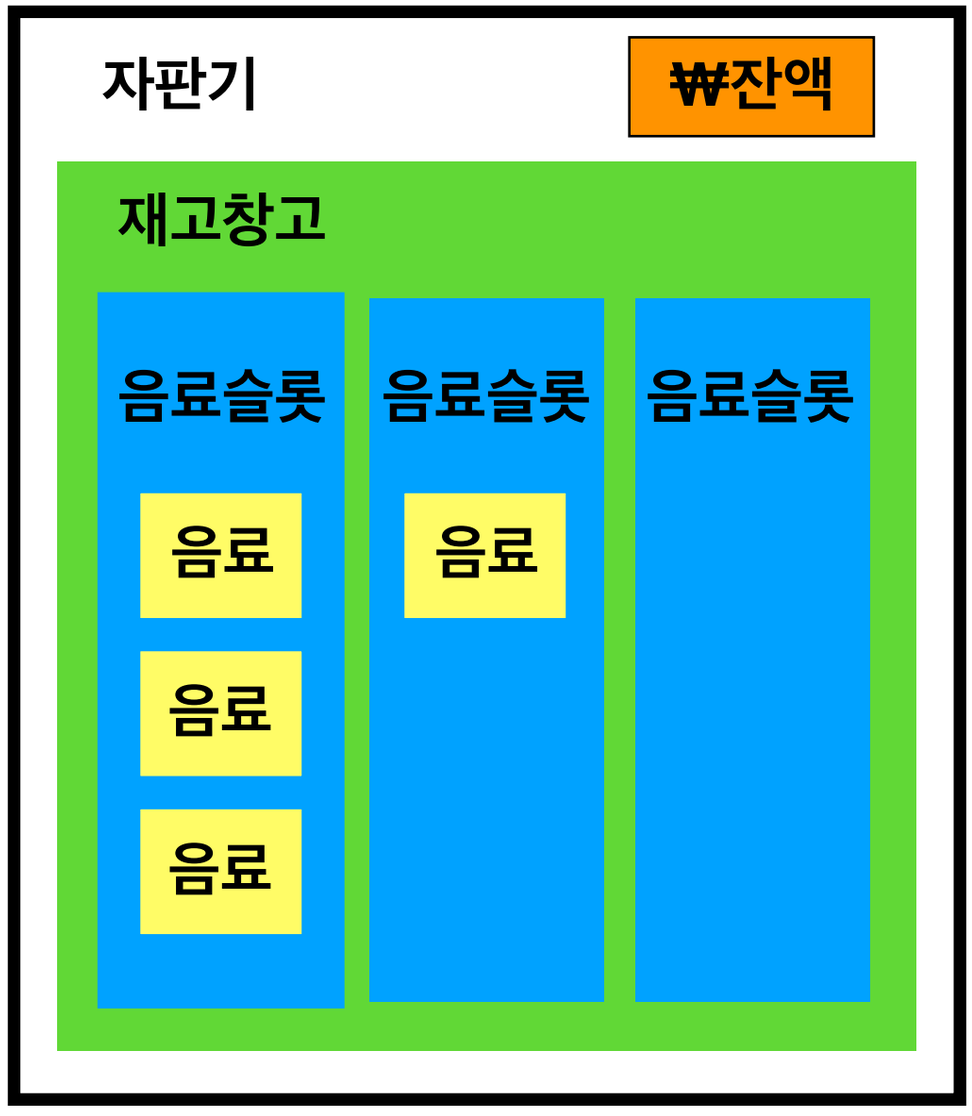
- 요구조건
- 자판기 구조
- 캔음료 객체 설계 과정
- 설계과정에서의 문제점
- 해결책

요구조건

- 자판기 객체를 만들어서 음료추가,구입 기능을 구현할것
- 자판기는 음료객체,잔액을 가진다.
- 음료는 이름,값,메이커,제조일자 등등의 데이터를 가진다.

자판기 구조

- 자판기는 잔액,재고창고를 소유
- 재고창고는 음료슬롯들을 소유
- 음료슬롯은 음료들을 소유
- 음료는 값,이름 등의 정보를 담음
- 현실의 자판기에 충실한 설계



캔음료 객체 설계 과정

- 이름, 가격, 브랜드 등의 변수 선언
- 주문할때 필요한 정보는 이름,가격 뿐
- 해당 정보를 가지는 음료정보 객체를 만들어서 데이터를 전달
- 음료의 모든 변수는 private 에 이름, 가격만 getter 생성

캔음료 객체 설계 과정

- 원하는 음료 추가 기능이 요구조건에 추가됨
- 원하는 음료를 추가 -> 모든정보가 같은 음료
- 음료의 모든 정보를 내보내야 가능
- 방법은 두가지.
    1. 모든 변수의 public 선언 - 안좋은 설계
    2. 모든 변수의 getter 추가 - 너무많은 함수추가

해결책

- 음료 객체의 설계가 꼭 현실과 같을필요는 없다!
  - 현실의 음료 정보를 얻기 위해선 음료 외부의 객체가 음료정보를 읽고, 해당 정보가 같은 음료를 추가
- 객체설계를 현실과 똑같이 할 경우 문제가 발생할수 있음
- 따라서 현실과 다르게 음료 객체에 **자가복제 함수를 추가**

해결책

```
class Drink : CustomStringConvertible {
    fileprivate let brand : String
    fileprivate let size : Int
    fileprivate let price : Int
    fileprivate let name : String
    fileprivate let manufacturingDate : Date
    let drinkType : DrinkInventory.DrinkType

-
/// 자기복제 함수
func duplicateSelf()->Drink{
    return Drink(barnd: self.brand, size: self.size, price: self.price, name: self.name
        , manufacturingDate: self.manufacturingDate, drinkType: self.drinkType)
}
```