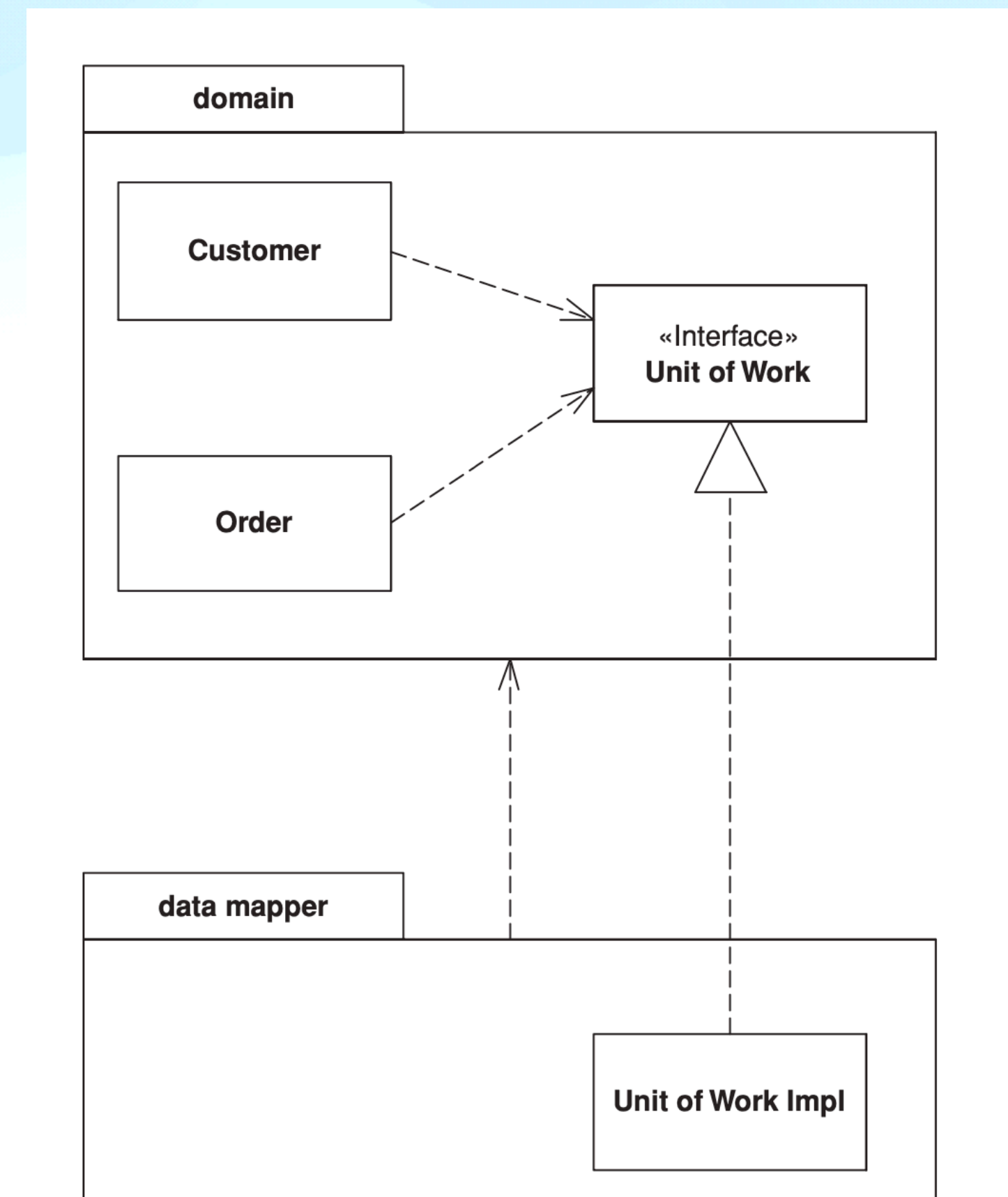
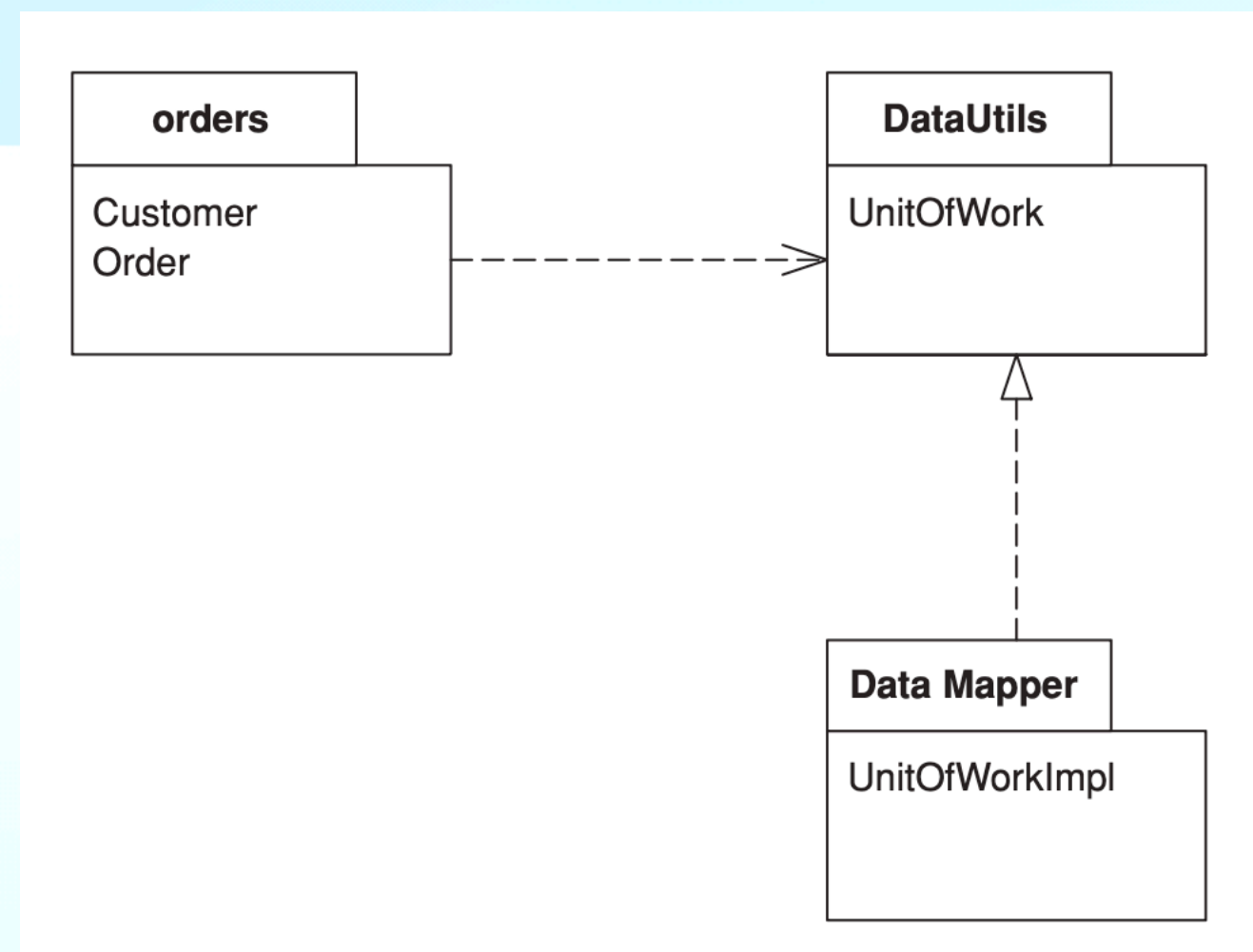


Separated Interface (Отделенный Интерфейс)

Виджая Стивен 932001

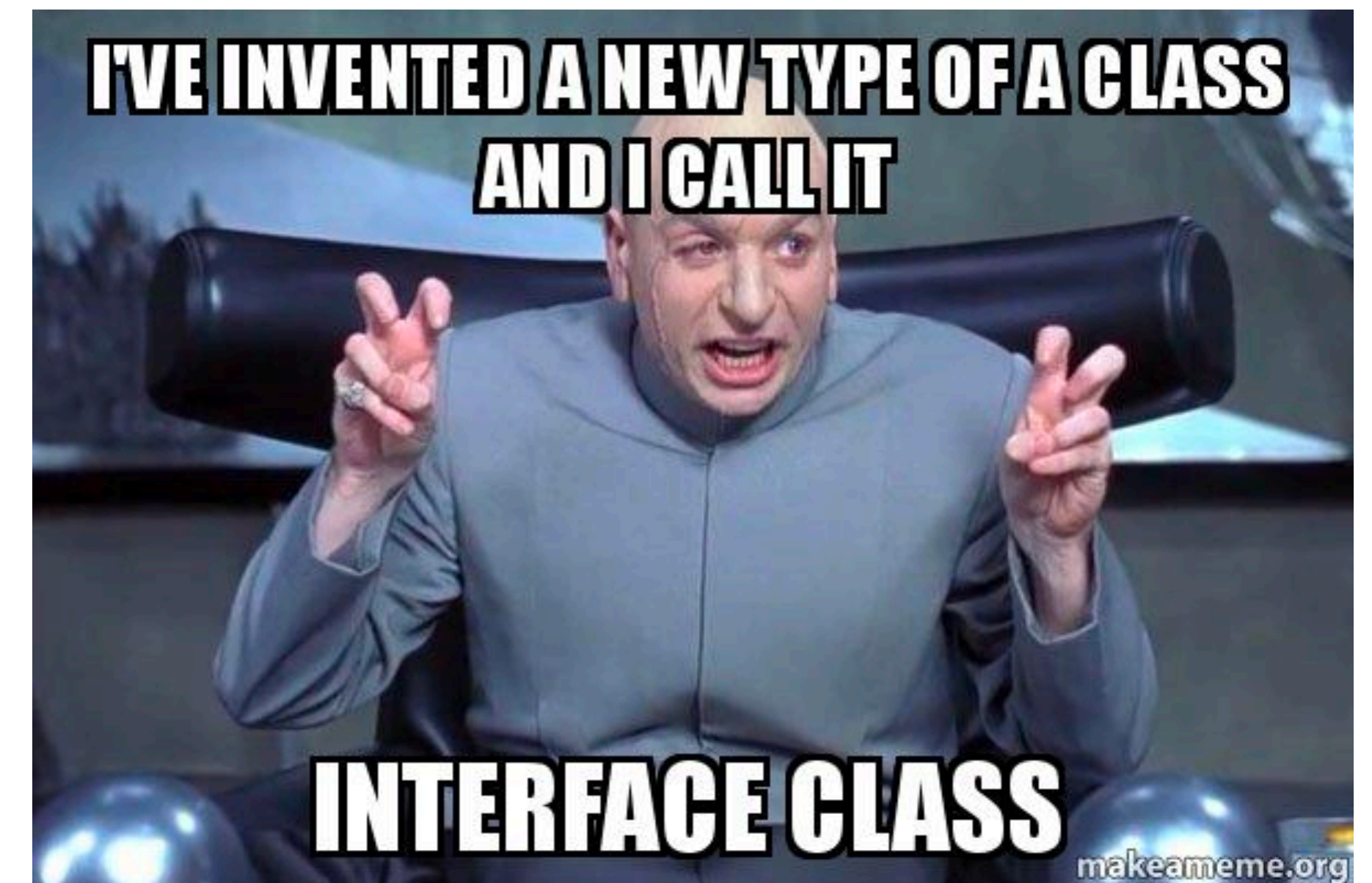
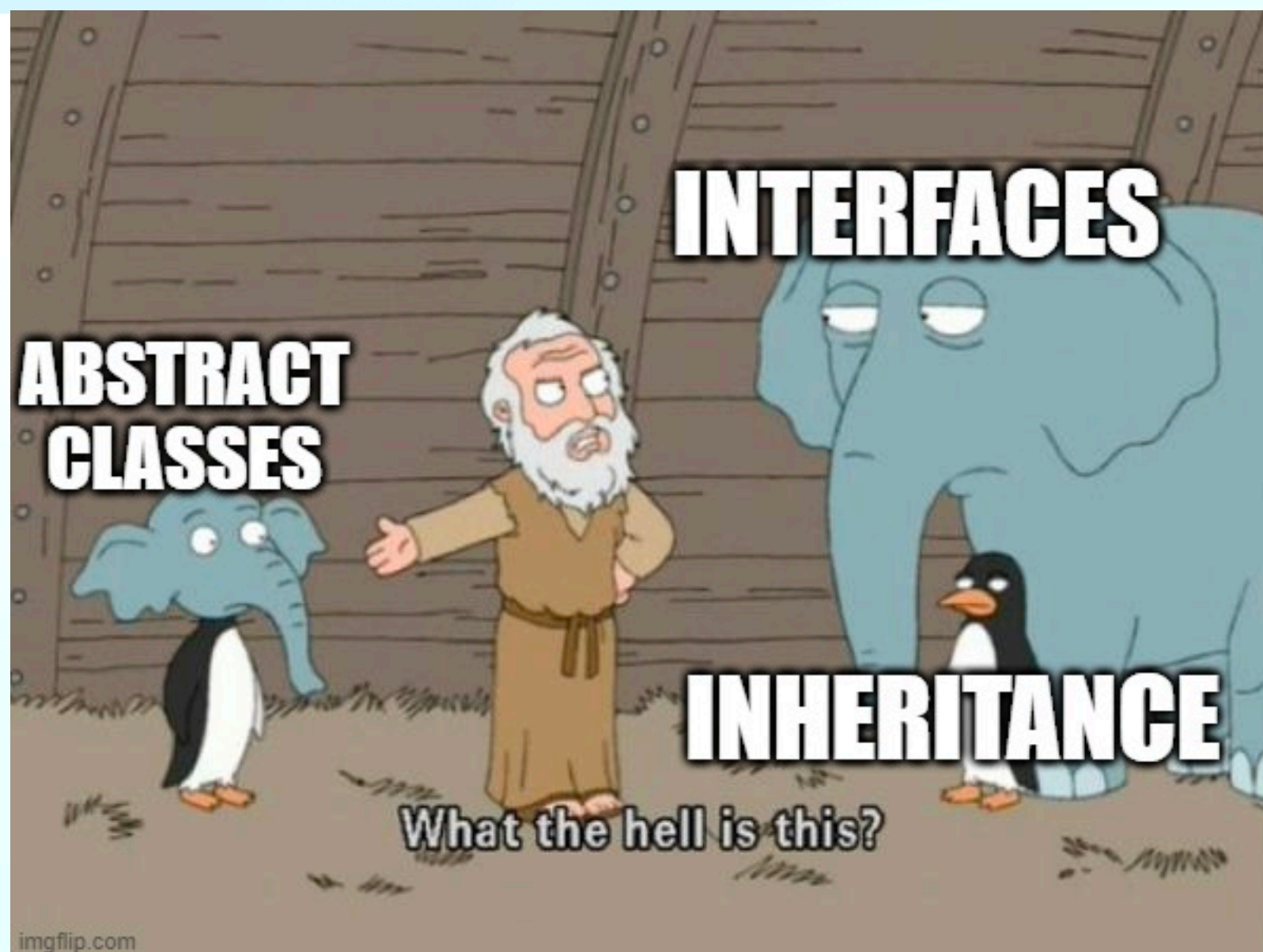
Определение

- Интерфейс в отдельном пакете от его реализации.
- Мы разделяем интерфейс от самой реализации. Интерфейс лежит в одном пакете а её реализация в другом.



Как паттерн работает?

- Он использует тот факт, что реализация зависит от своего интерфейса, но не наоборот. Это означает, что вы можете поместить интерфейс и реализацию в отдельные пакеты, а пакет реализации будет зависеть от пакета интерфейса. Другие пакеты могут зависеть от пакета интерфейса без зависимости от пакета реализации.



Когда нужно применить?

- Вы разрабатываете пакет фреймворка, и ваш фреймворк должен вызывать некоторый код приложения через интерфейсы.
- У вас есть отдельные пакеты, реализующие функциональные возможности, которые могут быть подключены к вашему клиентскому коду во время выполнения или во время компиляции.
- Ваш код находится на уровне, которому не разрешено вызывать уровень реализации интерфейса по правилу. Например, доменному слою необходимо вызвать средство сопоставления данных.

Пример

Interface

```
protocol ITaxCalculator {  
    var TaxPercentage: Double { get }  
    func calculate(price: Double) -> Double  
}
```

Implementation for TaxCalculator

```
class ForeignTaxCalculator: ITaxCalculator {  
    let TaxPercentage: Double = 45.0  
  
    func calculate(price: Double) -> Double {  
        price * TaxPercentage / 100.0  
    }  
}
```

Implementation for TaxCalculator

```
class DomesticTaxCalculator: ITaxCalculator {  
    let TaxPercentage: Double = 20.0  
  
    func calculate(price: Double) -> Double {  
        price * TaxPercentage / 100.0  
    }  
}
```


Пример

Объект Item (продукт)

```
struct Item: Identifiable {
    let id: UUID = UUID()
    var taxCalculator: ITaxCalculator
    var name: String
    var amount: Double
    var price: Double
    var tax: Double
    var finalPrice: Double

    init(taxCalculator: ITaxCalculator, name: String, amount: Double, price: Double) {
        self.taxCalculator = taxCalculator
        self.name = name
        self.amount = amount
        self.price = price
        self.tax = taxCalculator.calculate(price: self.price)
        self.finalPrice = self.price + self.tax
    }

    func getFinalPrice() -> Double {
        amount * (price + getTax())
    }

    func getTax() -> Double {
        round(taxCalculator.calculate(price: self.price) * 100) / 100.0
    }
}
```

Пример

Объект Invoice (для создания чека)

[illegible]

Пример

main

```
var invoice = Invoice()
invoice.addItem(
    Item(
        taxCalculator: Constant.domesticGoodsTax,
        name: "Russian Standard",
        amount: 35,
        price: 499.9
    )
)
invoice.addItem(
    Item(
        taxCalculator: Constant.importedGoodsTax,
        name: "Johnie Walker Blue Label",
        amount: 30,
        price: 18_000.00
    )
)
invoice.addItem(
    Item(
        taxCalculator: Constant.importedGoodsTax,
        name: "Egyptian Banana",
        amount: 1_000,
        price: 299.99
    )
)
print(invoice.generateInvoice())
```

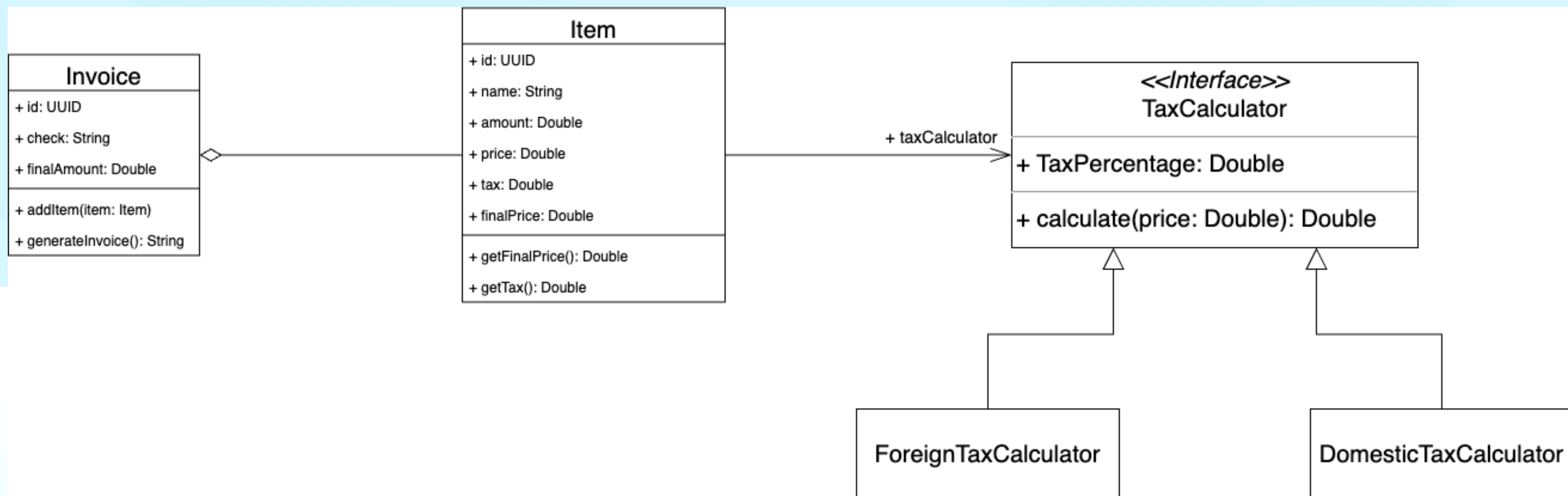
Для сохранения константных значений

```
struct Constant {
    static let importedGoodsTax: ITaxCalculator = ForeignTaxCalculator()
    static let domesticGoodsTax: ITaxCalculator = DomesticTaxCalculator()
}
```

Вывод в консоли

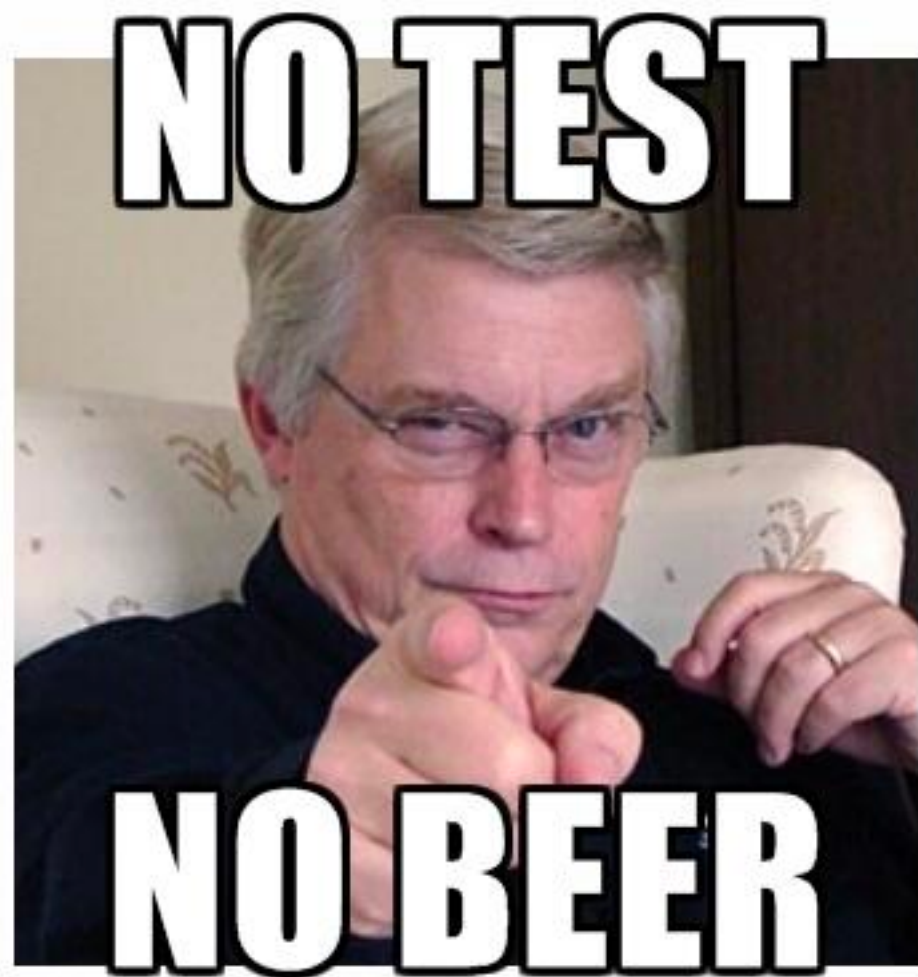
```
~      Invoice      ~
Russian Standard    499.9 + Tax: (20.0 %) 99.98 x 35.0
                                                           = 20995.8
Johnie Walker Blue Label    18000.0 + Tax: (45.0 %) 8100.0 x 30.0
                                                           = 783000.0
Egyptian Banana 299.99 + Tax: (45.0 %) 135.0 x 1000.0
                                                           = 434990.0
Final Amount:                                     = 1238985.8
```


Диаграмма Классов

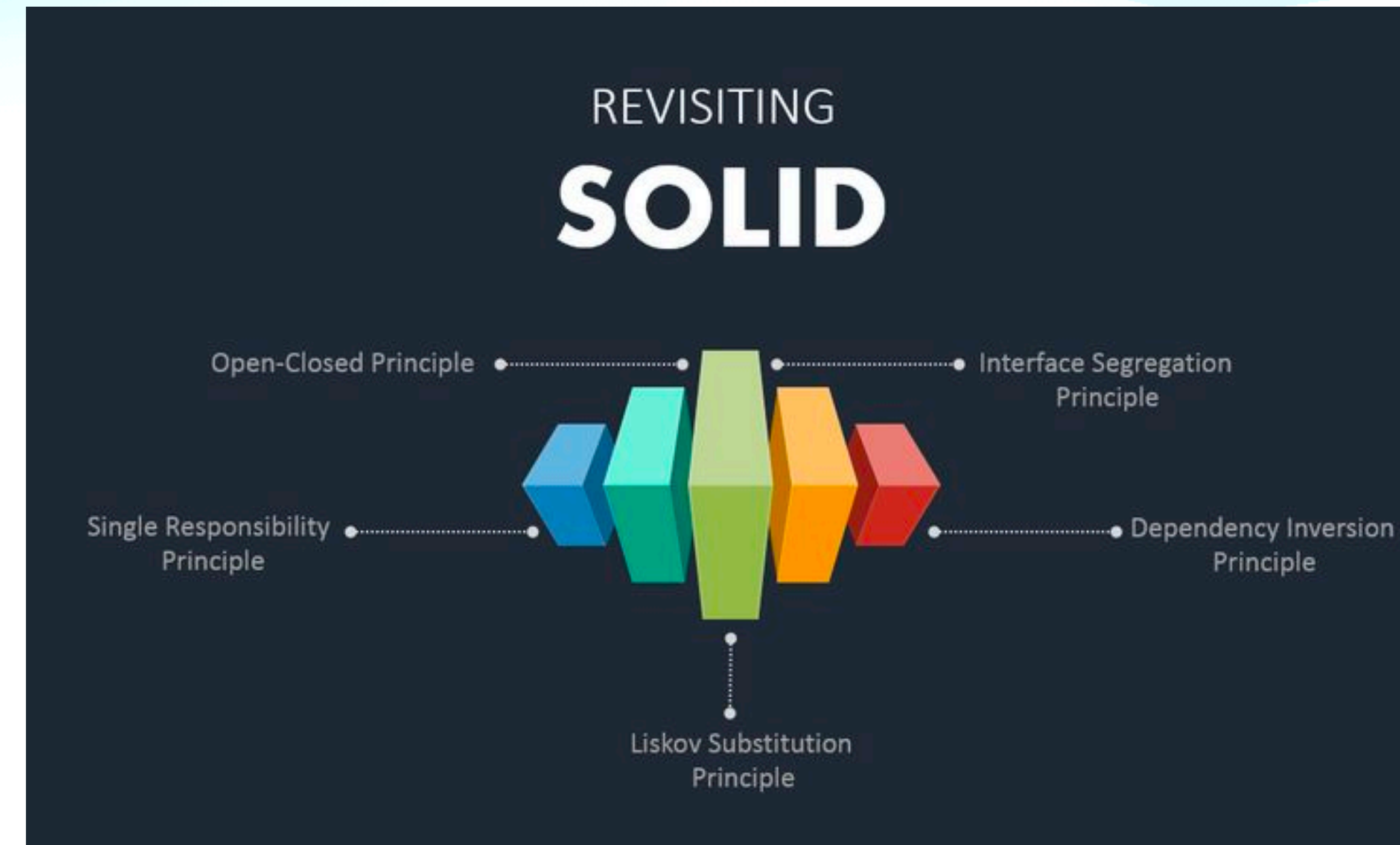


Interface Segregation Principle (ISP)

- Один из принципов SOLID
- “Клиентов не следует заставлять зависеть от интерфейсов, которые они не используют.”



memegenerator.co



Рассмотрим пример

<https://medium.com/movile-tech/interface-segregation-principle-in-swift-1778bab4452b>

Спасибо За Внимание

