

Введение в Swift

Лекция №2



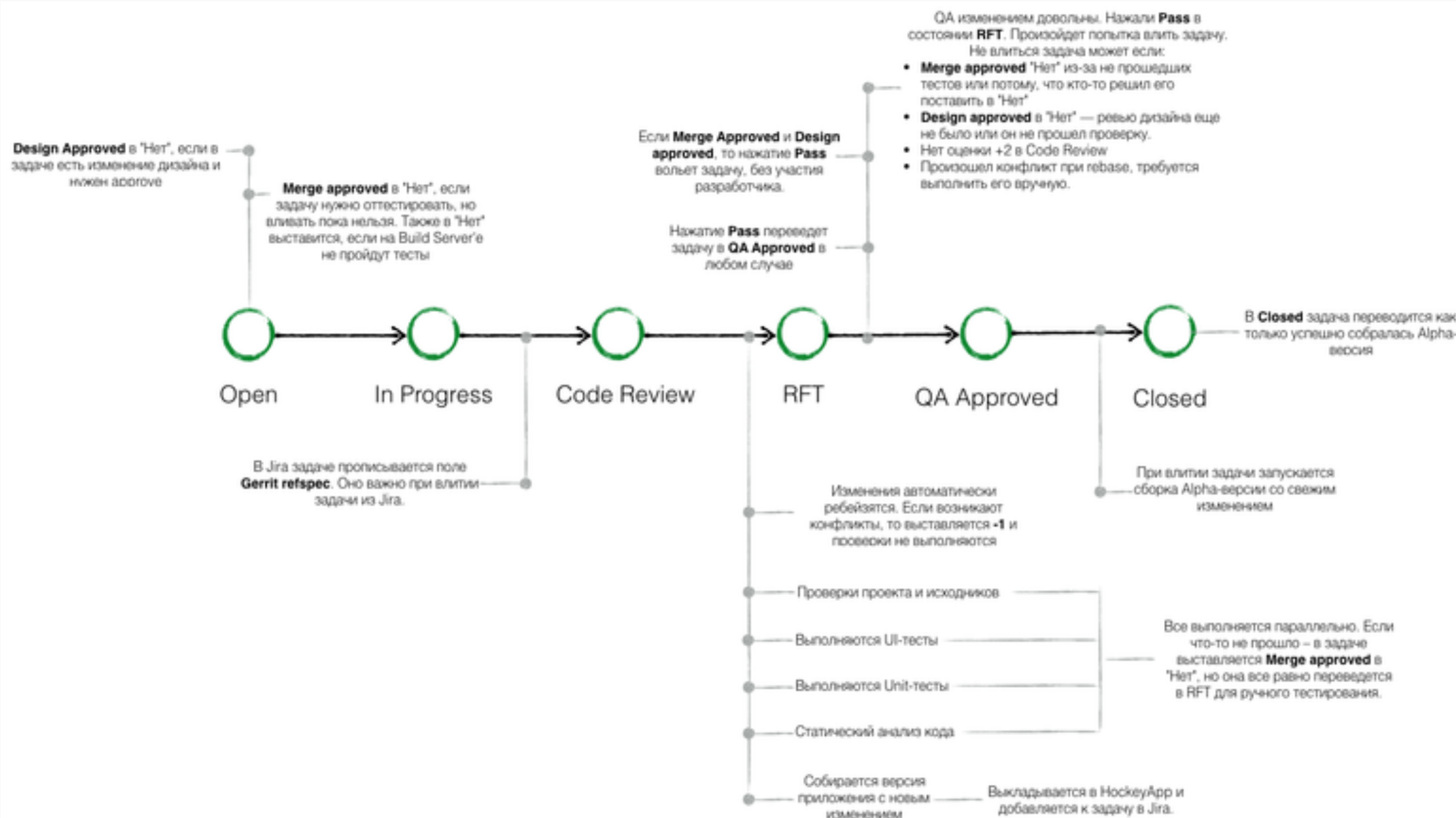
Андрей Решетников, Артур Сардарян,
Александр Каримов

Организационная часть



- Отметиться - важно
- О чем пойдет речь в сегодняшнем занятии
 - Процессы iOS почты
 - Swift
- Оставить отзыв (после занятия)

Процессы iOS почты



Домашнее задание



1. Придумать идею проекта
 - Расписать основные экраны
 - Общую информацию что и зачем
2. Разбиться на команды 2-3 человека
3. Создать репозиторий на гитхаб и выложить краткое описание

Дедлайн - конец 3-ей лекции

Структура лекции



- Swift
 - Модель памяти
 - Классы vs. структуры
 - Блоки
 - Протоколы
 - Расширения
 - Поточковая обработка коллекций

Константы и переменные



1. Константы объявляются с ключевым словом `let`
2. Переменные со словом `var`

Можно добавить обозначение типа

1. `var welcomeMessage: String`
2. `welcomeMessage = "Hello"`

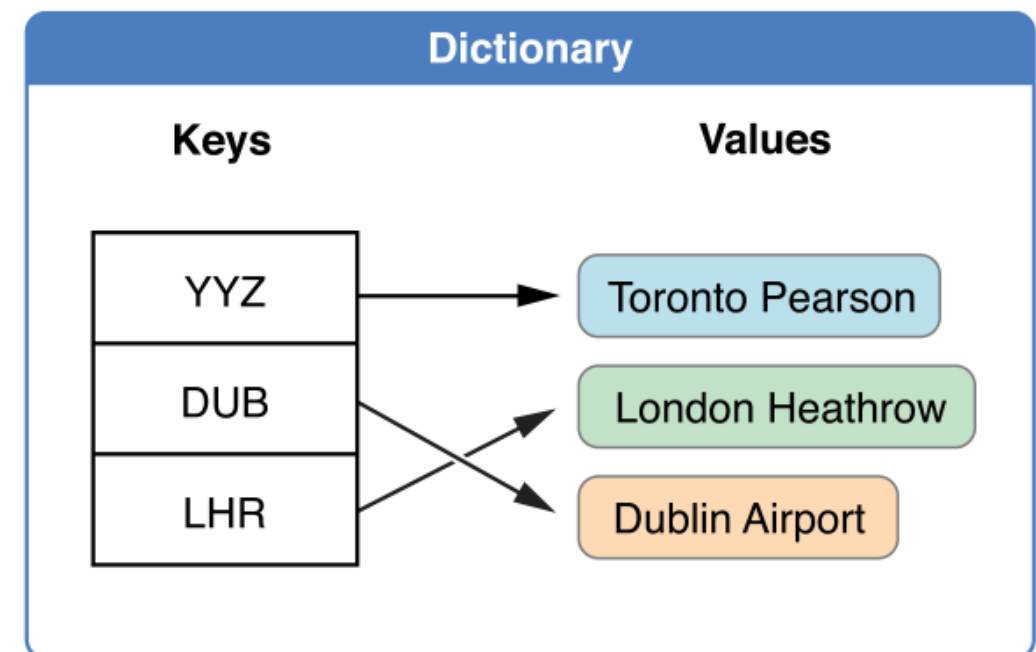
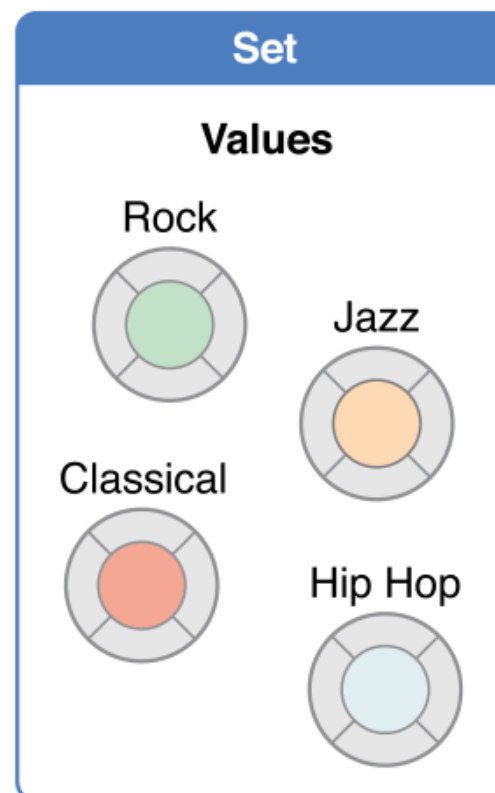
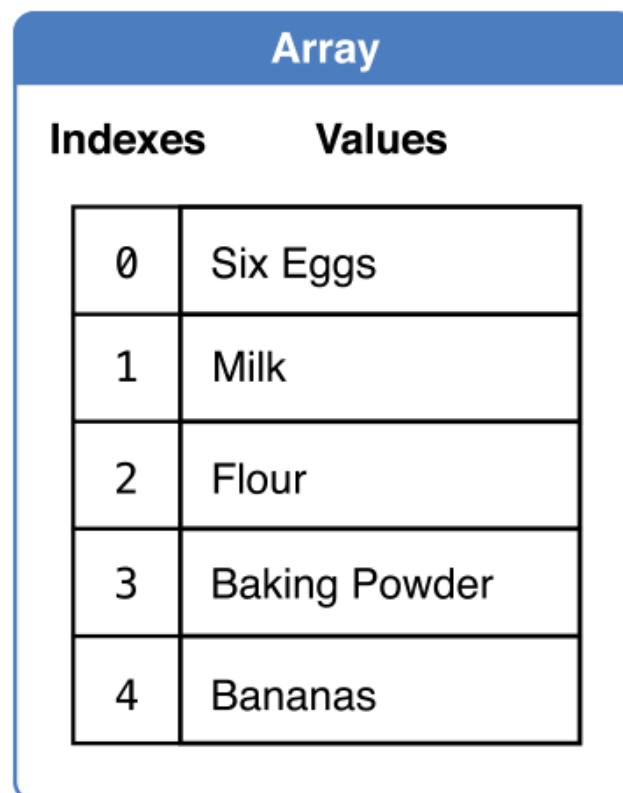
Swift предоставляет свои собственные версии фундаментальных типов C и Objective-C, включая **Int** для целых чисел, **Double** и **Float** для значений с плавающей точкой, **Bool** для булевых значений, **String** для текста. Swift также предоставляет мощные версии трех основных типов коллекций, **Array**, **Set** и **Dictionary**


```
let http404Error = (404, "Not Found")  
// http404Error имеет тип (Int, String), и равен (404,  
"Not Found")
```

Опциональные типы используются в тех случаях, когда значение может отсутствовать.

Опциональный тип подразумевает, что возможны два варианта: или значение есть, и его можно извлечь из опционала, либо его вообще нет.

1. Стандартные
2. Оператор `??` эквивалентен `a != nil ? a! : b`
3. Операторы диапазона `a...b`, `a..<b`, `a..., ...b`, `..`



1. `func greet(person: String) -> String {...}`
2. Можно вернуть несколько значений с помощью кортежа (a,b)
3. Есть сквозные параметры
4. Вложенные функции

1. Замыкания - это самодостаточные блоки с определенным функционалом, которые могут быть переданы и использованы в вашем коде. Замыкания в Swift похожи на блоки в C и Objective-C, и лямбды в других языках программирования.

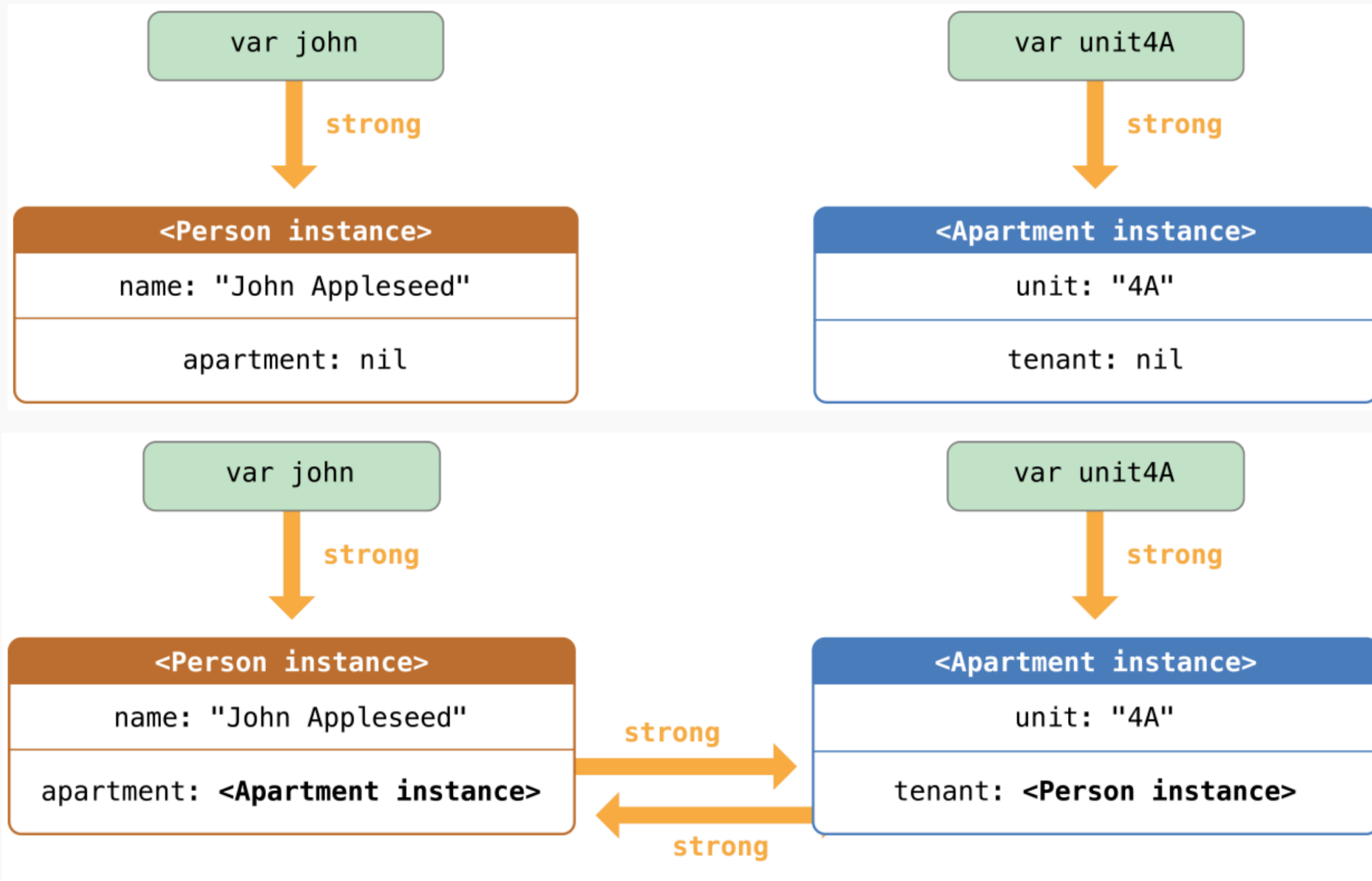
```
{ (параметры) -> тип результата in  
выражения  
}
```


Классы и структуры



```
class SomeClass {  
    // здесь пишется определение класса  
}  
  
struct SomeStructure {  
    // здесь пишется определение структуры  
}
```

Swift использует automatic reference counting (автоматический подсчет ссылок) для отслеживания и управления памятью вашего приложения. ARC автоматически освобождает память, которая использовалась экземплярами класса, когда эти экземпляры больше нам не нужны.



var john

var unit4A

<Person instance>

name: "John Appleseed"

apartment: **<Apartment instance>**

strong

<Apartment instance>

unit: "4A"

tenant: **<Person instance>**

strong

1. <https://swift.org/>
2. <https://swiftbook.ru/content/languageguide/>
3. <https://itunes.apple.com/ru/course/developing-ios-11-apps-with-swift/id1309275316>