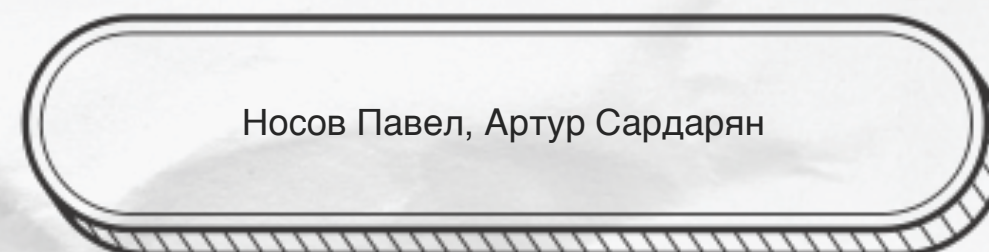


Лекция №2



Организационная часть



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

Структура лекции



- Hello world!
- Константы и переменные, типизация
- Операторы
- Кортежи
- Опционалы
- Коллекции
- Функции
- Замыкания
- Классы vs. структуры
- Память

А так же:

Блоки

Протоколы

Расширения

Историческая справка



Swift – был представлен компанией Apple в 2014 году в первую очередь для разработчиков iOS/macOS.

Objective-C

```
const int count = 10;
double price = 23.55;

NSString *firstMessage = @"Swift is awesome. ";
NSString *secondMessage = @"What do you think?";
NSString *message = [NSString stringWithFormat:@"%s%s", firstMessage, secondMessage];

NSLog(@"%s", message);
```

Swift

```
let count = 10
var price = 23.55

let firstMessage = "Swift is awesome. "
let secondMessage = "What do you think?"
var message = firstMessage + secondMessage

print(message)
```


Hello world!



```
1 print("Hello, world!")
2 // Prints "Hello, world!"
```

<— Эта строчка кода – полноценная программа

```
4
5 var text = "Hello, World"
6 print(text)
7
```

<— тоже самое, только с переменной

Константы и переменные



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

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**

Можно добавить обозначение типа

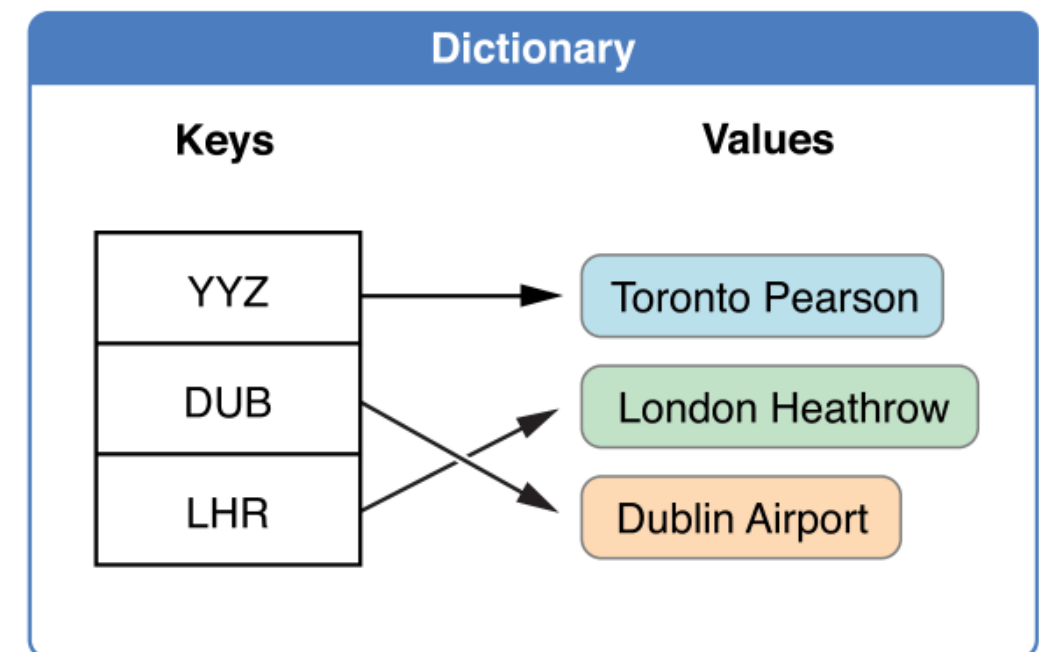
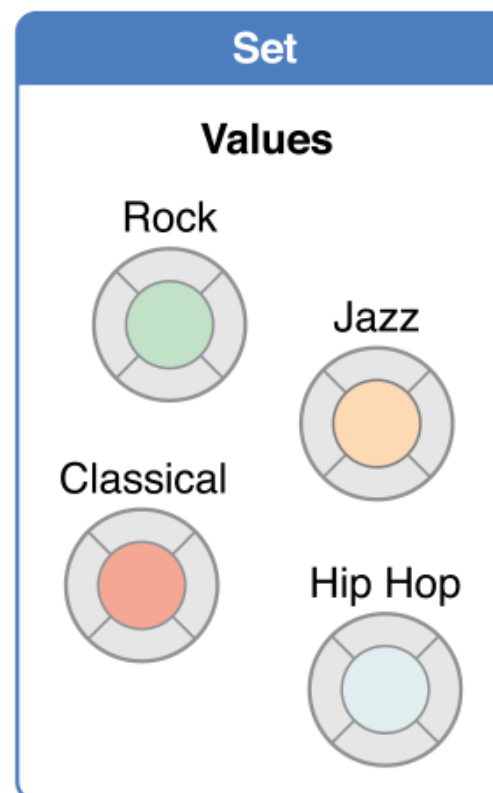
1. `var welcomeMessage: String`
2. `welcomeMessage = "Hello"`


```
let http404Error = (404, "Not Found")  
// http404Error имеет тип (Int, String), и равен (404,  
"Not Found")  
print(http404Error.0)  
// Выведет 404
```

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`, `..`

Array	
Indexes	Values
0	Six Eggs
1	Milk
2	Flour
3	Baking Powder
4	Bananas



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  
}
```

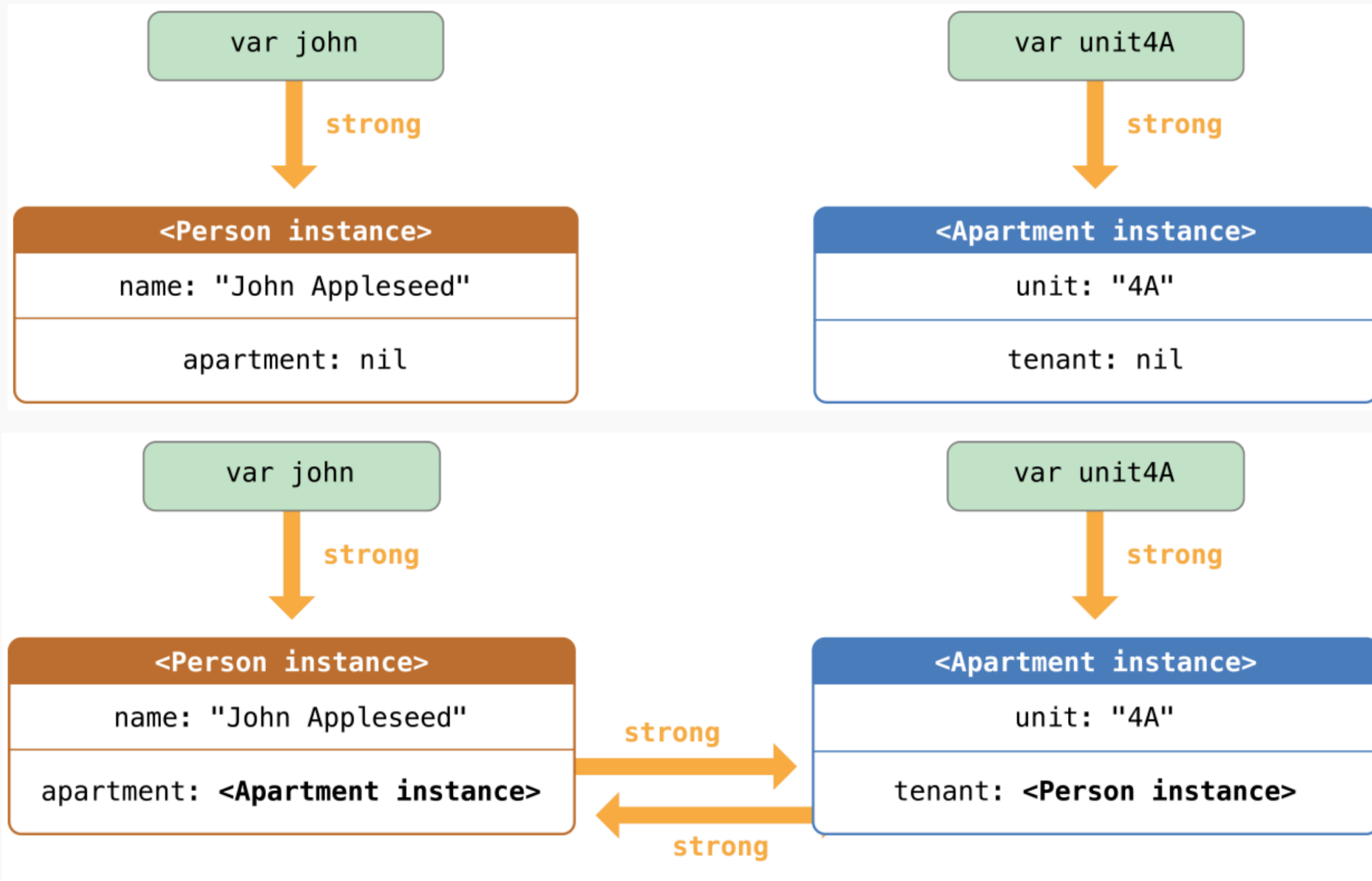

Классы и структуры



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

Swift использует automatic reference counting (автоматический подсчет ссылок) сокр. ARC 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

Домашнее задание



1. Придумать идею проекта
 - Расписать основные экраны
 - Общую информацию что и зачем
2. Разбиться на команды до 4 человек
3. Создать репозиторий на GitHub и выложить краткое описание в readme

Срок: конец 3-ей лекции

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