



Работа с Xcode

Лекция №3



Андрей Решетников, Артур Сардарян,
Александр Каримов

Организационная часть



- Самое интересное - отметиться
- О чём пойдет речь в сегодняшнем занятии
 - Обзор Xcode
 - Окошечки
 - Проект, таргет
 - Форматы файлов
 - Процесс запуска приложения
 - Interface Builder
 - Autolayout
 - Переходы между экранами
 - Swift vs Objective-C
- Оставить отзыв (после занятия)

Создание проекта



**Уникальный
Bundle identifier**

Choose options for your new project:

Product Name:

Team:

Organization Name:

Organization Identifier:

Bundle Identifier:

Language:

User Interface:

Use Core Data

Use CloudKit

Include Unit Tests

Include UI Tests

Xcode



Toolbar

Navigator Area

Editor Area

Utility Area

Debug Area

The image shows the Xcode interface with several areas highlighted by a red border:

- Navigator Area**: The leftmost column containing the Project Navigator, Document Outline, and Utilities.
- Editor Area**: The central workspace where project settings and code are edited.
- Utility Area**: The rightmost column containing the Utilities panel.
- Debug Area**: The bottom section showing the Debug Navigator and the Output pane.

The Editor Area displays the "General" tab of the TestApp target settings. It includes sections for Identity (Display Name: TestApp, Bundle Identifier: ru.mail.TestApp), Deployment Info (Target: iPhone, iPad; Main Interface: Main), and App Icons and Launch Images (App Icons Source: AppIcon, Launch Screen File: LaunchScreen). The Utilities panel on the right shows the Identity and Type, Project Document, and Text Settings sections. The bottom Debug Area shows the output "Hey there! (lldb)" from the debugger.

Playground



The screenshot shows the Xcode playground interface. On the left, the **Swift code Editor** contains the following Swift code:

```
1 import Foundation
2
3 var magicNumber = 1
4
5 magicNumber += 3
6 magicNumber /= 2
7 magicNumber *= magicNumber
8
9 print("Magic number = \(magicNumber)")
```

A play button is visible below the code editor.

In the center, the **Results Live view** displays the output of the code execution:

1	
4	
2	
4	
"Magic number = 4\n"	

At the bottom, the **Debug Area** shows the output "Magic number = 4".

Playground



Зачем нужен Playground?

Проект



- Включает в себя все ресурсы и файлы
- Содержит информацию о сортировке и группировке файлов
- Хранит настройки и описание таргетов



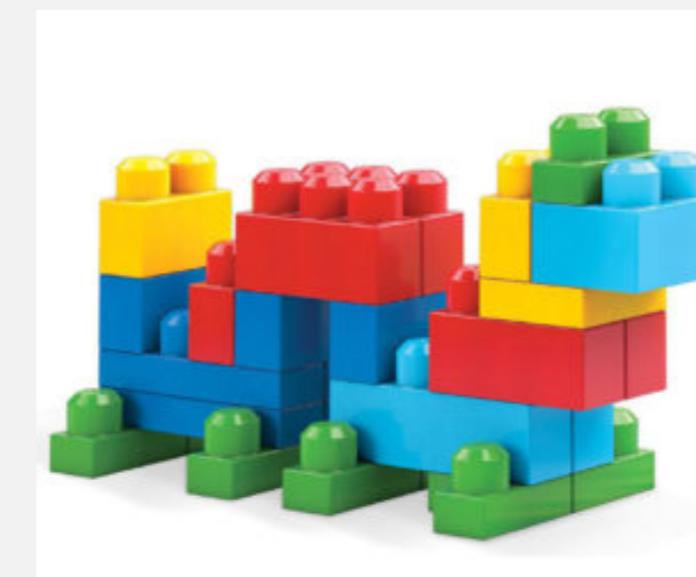
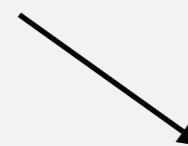
Таргет



- Хранит список файлов для сборки
- Содержит настройки и фазы сборки



Проект, таргет



Форматы файлов



- *.m, *.h - файл с Objective-C кодом
- *.swift - файл с Swift кодом
- *.xib - XML с UI элементами
- *.storyboard - XML с экранами и переходами
- *.plist - файл с настройками проекта
- *.xcassets - папка с картинками
- *.playground - ?



Запуск приложения



**Что происходит по нажатию
на иконку приложения?**

Запуск приложения



UIApplicationMain



```
9 import UIKit  
10  
11 @UIApplicationMain  
12 class AppDelegate: UIResponder, UIApplicationDelegate {
```

- Создает объект приложения, настраивает цикл обработки событий

Declaration

```
func UIApplicationMain(_ argc: Int32,  
                      _ argv: UnsafeMutablePointer<UnsafeMutablePointer<Int8>?>,  
                      _ principalClassName: String?,  
                      _ delegateClassName: String?) -> Int32
```

UIApplication



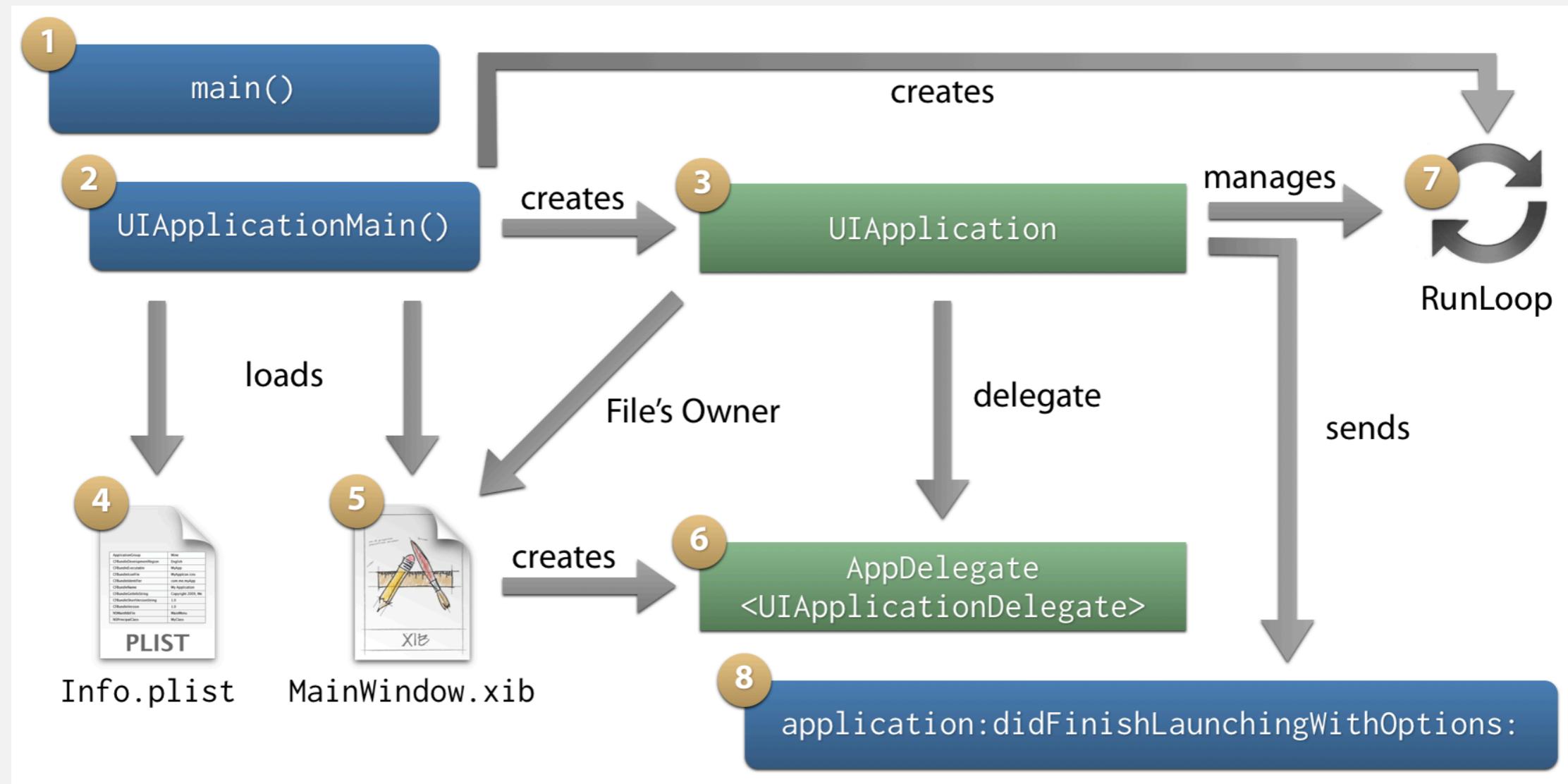
- “The centralized point of control and coordination for apps running in iOS.” - центральная точка управления приложением
- Синглтон
- Принимает пользовательские события, передает их UI компонентам
- Управляет уведомлениями (remote notifications)
- Умеет запускать фоновые задачи
- Открывает ссылки на внешние источники или в другие приложения (URL scheme)
- Тригер на ShakeToEdit

UIApplicationDelegate

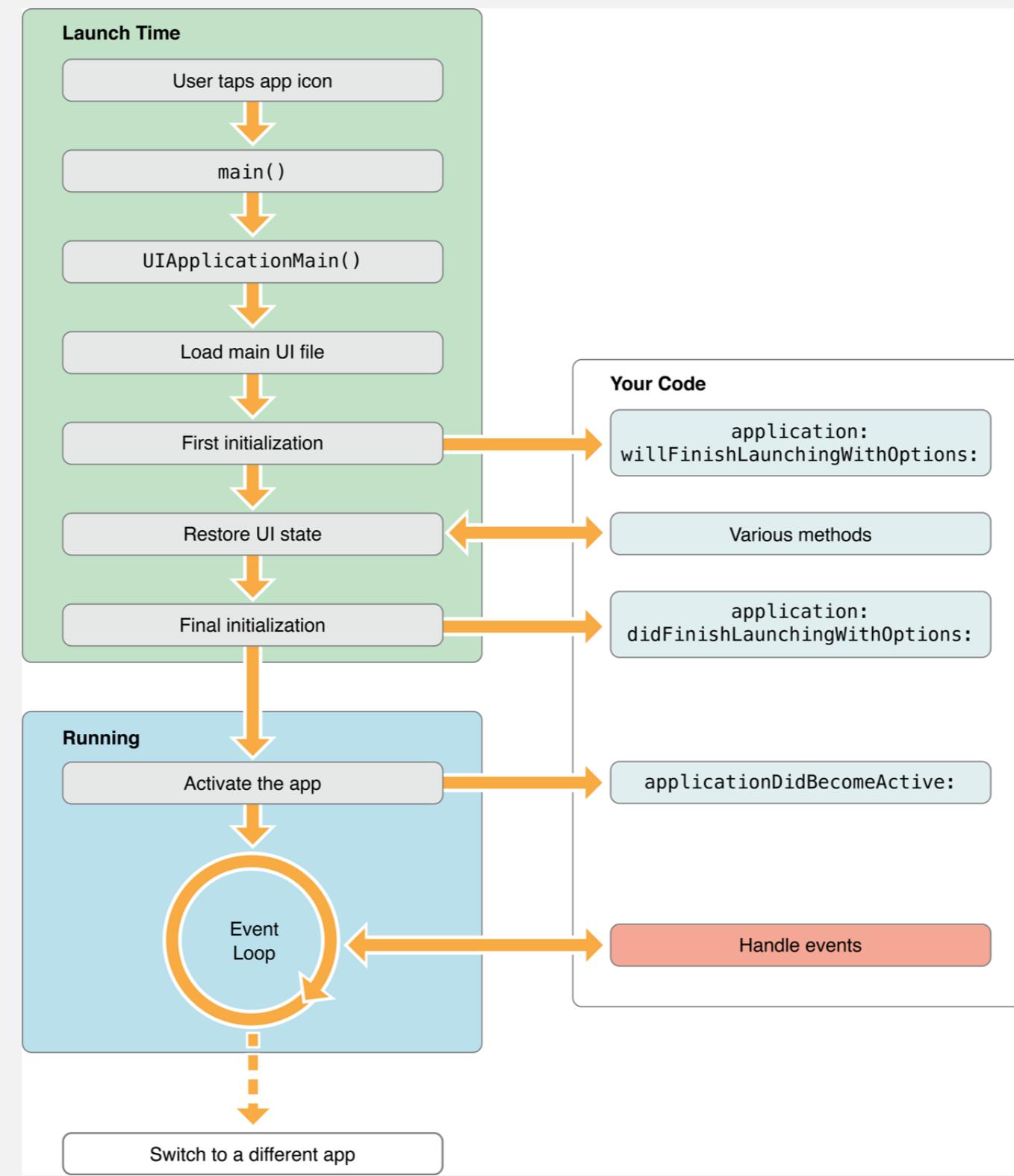


- Всегда существует (создается при запуске)
- Обрабатывает события жизненного цикла приложения
- Обрабатывает внешние события

Launch sequence



Launch sequence



Swift vs Objective-C



Swift vs Objective-C



Плюсы Objective-C:

- Динамическая типизация (в некоторых случаях ускоряет разработку)
- Прекрасно работает совместно с языком C
- Разные хаки (method swizzling)

Method swizzling

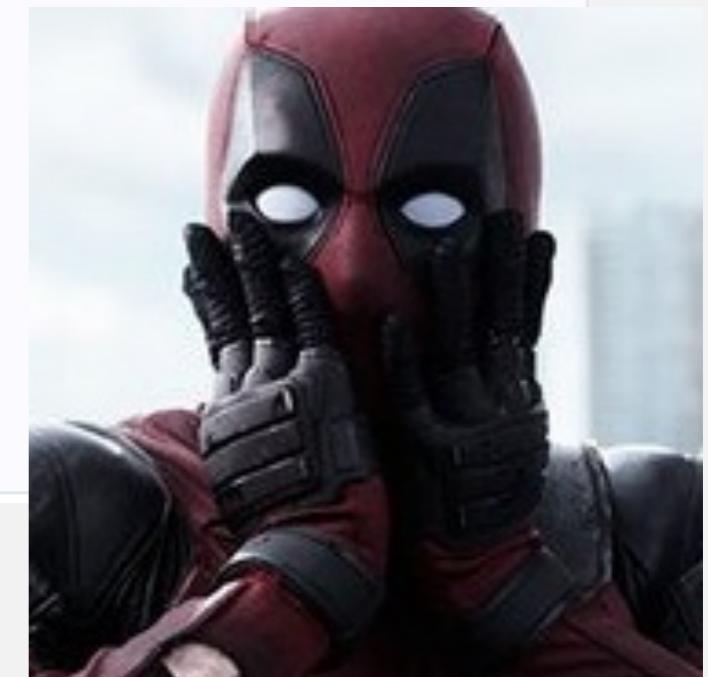


```
@implementation NSMutableArray (CO)

+ (void)load {
    Method addObject = class_getInstanceMethod(self, @selector(addObject:));
    Method logAddObject = class_getInstanceMethod(self, @selector(logAddObject:));
    method_exchangeImplementations addObject, logAddObject;
}

- (void)logAddObject:(id)aObject {
    [self logAddObject:aObject];
    NSLog(@"Добавлен объект %@ в массив %@", aObject, self);
}

@end
```



Swift vs Objective-C



Минусы Objective-C:

- Динамическая типизация (пропуск ошибок во время компиляции)
- Плохая читаемость (особенно новичкам)
- Явные указатели
- Возможность отправить сообщение nil без крэша (сложнее поймать баг)
- Ограниченнная функциональность (нет модных штук, как в Swift)

Swift vs Objective-C



```
var array = [Int]()
```

```
NSMutableArray *array = [[NSMutableArray alloc] init];
```

```
let str = "Hello World"  
print(str)
```

```
NSString *str = @"Hello World";  
NSLog(@"%@", str);
```

Swift vs Objective-C



Плюсы Swift:

- Безопаснее благодаря строгой типизации и опционалов
- Поддерживает namespace (вложенные классы)
- Есть playground
- Продолжает развиваться
- Полноценное взаимодействие с Objective-C

Swift vs Objective-C



Минусы Swift:

- Страдает время компиляции (боль в проектах где и obj-C, и swift)
- Нет прямого способа использовать библиотеки на C/C++
- Компилятор выдает слишком много ошибок

Live Coding



- Верстка
- Переходы: segue vs present/push

Ссылки



- <https://www.raywenderlich.com/443-auto-layout-tutorial-in-ios-11-getting-started> - Autolayout
- <https://developer.apple.com/documentation/uikit/uiapplication> - UIApplication
- <https://codewithchris.com/xcode-tutorial/> - про Xcode
- <https://developer.apple.com/documentation/uikit/uiviewcontroller> - жизненный цикл UIViewController
- <https://park.mail.ru/blog/topic/view/12045/> - жизненный цикл приложения
- <https://developer.apple.com/library/archive/featuredarticles/ViewControllerPGforiPhoneOS/UsingSegues.html> - про segues
- <https://github.com/raywenderlich/swift-style-guide> - code style
- <https://medium.com/roberryapps/ios-safe-area-ca10e919526f> - safe area