# ЖИЗНЕННЫЙ ЦИКЛ ПРИЛОЖЕНИЯ В ОПЕРАЦИОННОЙ СИСТЕМЕ IOS

APPLICATION LIFECYCLE IN THE IOS OPERATING SYSTEM

**УДК 004.4**

**Шапетько Л.А.,** студент, 4 курс, Институт информационных технологий, МИРЭА – Российский Технологический Университет Россия, г. Москва

**Носов В.Р.,** студент, 4 курс, Институт информационных технологий МИРЭА – Российский Технологический Университет Россия, г. Москва

**Шушарин М.А.,** студент, 4 курс, Институт информационных технологий, МИРЭА – Российский Технологический Университет Россия, г. Москва

**Shapetko L. A.,** shapetko.leonid@gmail.com

**Nosov V. R.,** shapetko.leonid@gmail.com

**Shusharin M. A.,** shapetko.leonid@gmail.com

**Аннотация**

При разработке мобильного приложения может возникнуть необходимость в реализации методов, которые функционируют на различных этапах жизненного цикла приложения. В данной статье анализируются модель жизненного цикла, обозначаются основные этапы жизненного и описываются методы, используемые на каждом этапе. Для каждого метода дается краткое описание. На практическом примере показываются функционирование основных методов модели.

**Annotation**

When developing a mobile application, it may be necessary to implement methods that function at different stages of the application lifecycle. This article analyzes the life cycle model, identifies the main stages of the life cycle, and describes the methods used at each stage. A brief description is given for each method. A practical example shows the functioning of the main methods of the model.

**Ключевые слова:** iOS, жизненный цикл приложения, разработка мобильных приложений, AppDelegate, SceneDelegate.

**Keywords:** iOS, App Lifecycle, Mobile App Development, AppDelegate, SceneDelegate.

**Постановка проблемы.** В процессе разработки мобильных приложения для операционной системы iOS разработчик может столкнуться с необходимостью реализации функционала, выполняемого на определенных этапах жизненного цикла приложения – к примеру, во время его запуска и до загрузки экранов графического интерфейса или в момент, когда пользователь выходит из приложения. Для реализации необходимого функционала могут быть использованы методы протокола UISceneDelegate, который предоставляет большое количество методов, исполняемых приложением на различных этапах жизненного цикла. Знание данных методов может помочь разработчику сделать функционирование программы более гибким.

**Целью статьи** является описание модели жизненного цикла приложения в операционной системе iOS, описание основных состояний жизненного цикла и методов, используемых на данных этапах. Кроме этого, на практических примерах будут продемонстрировано применение основных методов модели жизненного цикла приложения.

**Изложение основного материала.** До релиза iOS 13 основным объектом, который отвечал за жизненный цикл приложения, был объект AppDelegate, который был подписан на протокол UIApplicationDelegate. Данный протокол предоставляет методы, выполняемые на различных этапах жизненного цикла приложения [1]. В книге Ванга В. «Профессиональная разработка для iPhone на языке Swift 5» (Wang W. (2019). Pro iPhone Development with Swift 5) данной модели отведена отдельная глава с подробным разбором как самой концепции, так и используемых в ней методов.

С релизом iOS 13 произошли некоторые изменения – была добавлена поддержка многооконных приложений для iPad, в результате чего часть методов жизненного цикла, использовавшиеся в UIApplicationDelegate, были

перемещены в протокол UISceneDelegate и его объект SceneDelegate. Стоит отметить, что даже с учетом данных изменений жизненный цикл для мобильных приложений для iPhone (и, тем самым, для iOS) не претерпел кардинальных изменений, так как приложения для iOS используют только лишь одно активное окно-сцену. В данной статье будет рассматриваться модель жизненного цикла, которая появилась с iOS 13.

Этапы жизненного цикла [2]:

- приложение будет запущено. В данное состояние приложение переходит после нажатия на иконку приложения, но до загрузки основных объектов. При переходе на данный этап вызывается метод application (_ application: UIApplication, willFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]? = nil) -> Bool [3] объекта AppDelegate. Данный метод можно переопределить для выполнение специфических задач во время загрузки приложения.

- приложение было запущено. В данное состояние приложение переходит после загрузки основных объектов, но до загрузки объектов графического интерфейса. При переходе на данный этап вызывается метод application (_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey : Any]? = nil) -> Bool [4] объекта AppDelegate. Данный метод можно переопределить для выполнения специфических задач после загрузки основных объектов приложения.

- загрузка окна графического интерфейса приложения в память устройства. В данное состояние приложение переходит после загрузки основных объектов и начала инициализации и привязки окна, на котором будут отображаться экраны графических интерфейсов. При переходе на данный этап вызывается метод application (_ application: UIApplication, configurationForConnecting connectingSceneSession: UISceneSession, options: UIScene.ConnectionOptions) -> UISceneConfiguration [5] объекта AppDelegate. Данный метод можно переопределить для выполнения специфических задач после начала загрузки окна и его конфигурации или если требуется изменять параметры окна в процессе выполнения приложения [5].

- окно графического интерфейса должно стать активным. В данное состояние приложение переходит после загрузки окна графического интерфейса в память и перед его отображением на экране устройства. При переходе на данный этап вызывается метод sceneWillEnterForeground(_ scene: UIScene) [6] объекта SceneDelegate. Данный метод можно переопределить для выполнения специфических задач после загрузки окна в память, но до его непосредственного отображения.

- окно графического интерфейса стало активным. В данное состояние приложение переходит после завершения процесса отображения окна, тем сама предоставляя возможность пользователю взаимодействовать с элементами графического интерфейса. При переходе на данный этап вызывается метод sceneDidBecomeActive(_ scene: UIScene) [7] объекта SceneDelegate. Данный метод можно переопределить для выполнения специфических задач

- окно графического интерфейса должно стать неактивным. В данное состояние приложение переходит при необходимости временно прервать исполнение приложения и убрать окно с экрана устройства – к примеру, при поступающем на устройство звонке. При переходе на данный этап вызывается метод sceneWillResignActive (_ scene: UIScene) [8] объекта SceneDelegate. Данный метод можно переопределить для выполнения специфических задач

- окно графического интерфейса было убрано. В данное состояние приложение переходит в случае, когда окно было убрано с экрана, но не выгружено из памяти. При переходе на данный этап вызывается метод sceneDidEnterBackground (_ scene: UIScene) [9] объекта SceneDelegate. Данный метод можно переопределить для выполнения специфических задач при переходе приложения в «неактивное» состояние – к примеру, сохранение всех данных приложения.

- окно графического интерфейса было выгружено. В данное состояние приложение переходит только в том случае, если если операционной системе требуются дополнительные аппаратные ресурсы для исполнения другого приложения. При переходе на данный этап вызывается метод sceneDidDisconnect (_ scene: UIScene) [10] объекта SceneDelegate. Данный метод можно

переопределить для выполнения специфических задач перед выгрузкой основного окна графического интерфейса из памяти.

- приложение было завершено. В данное состояние приложение переходит в случае, если его работа завершается. При переходе на данный этап вызывается метод applicationWillTerminate (_ application: UIApplication) [11] объекта AppDelegate. Данный метод можно переопределить для выполнения специфических задач перед полным завершением приложения, однако предпочтительнее реализовать исполнение методов на этапах, идущих перед этапом завершения приложения.

Диаграмма этапов жизненного цикла с указанием вызываемых методов при переходе из одного этапа в другой показана на рисунке 1 [2].




**Рисунок 1 – Диаграмма жизненного цикла приложения в операционной системе iOS**

## Практическое применение методов жизненного цикла

Рассмотрим на практическом примере как функционируют вышеописанные методы. В качестве примера была разработана программа в среде разработки Xcode с использованием встроенного редактора графических интерфейсов Interface Builder. Версия iOS, используемая в эмуляторе устройства – 14.2.

Разработанная программа на этапе подключения сцены к приложению отправляет запрос к серверу на проверку статуса подключения, после чего происходит загрузка начального экрана, содержащего элементы графического –

два текстовых поля для ввода логина и пароля и кнопку, функционал которой меняется в зависимости от статуса подключения. В зависимости от статуса подключения меняется интерфейс начального экрана – кроме изменения текста кнопки отображается текстовое сообщение ошибки подключения. Кроме этого, на каждом этапе жизненного цикла методы выводят сообщения со своим названием в консоль среды разработки. Код используемого ViewController представлен в листинге 1, код AppDelegate – в листинге 2, код SceneDelegate – в листинге 3.

Листинг 1.

Описание используемого ViewController

```
var connectionStatus: Bool = true

class ViewController: UIViewController {

    @IBOutlet weak var usernameTF: UITextField!
    @IBOutlet weak var passwordTF: UITextField!
    @IBOutlet weak var statusLabel: UILabel!
    @IBOutlet weak var actionButton: UIButton!

    override func viewDidLoad() {
        super.viewDidLoad()
        if connectionStatus == false {
            statusLabel.text = "Connection Error, please try
again"
            actionButton.setTitle("Check connetcion", for:
.normal)
        } else {
            statusLabel.isHidden = true
            actionButton.setTitle("Enter", for: .normal)
        }
    }
}
```

Листинг 2.

Описание используемого AppDelegate

```
@main
class AppDelegate: UIResponder, UIApplicationDelegate {

    func application(_ application: UIApplication,
willFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey : Any]? = nil) -> Bool {
        print (#function)
        return true
    }
```

```
    func application(_ application: UIApplication,
didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]?) -> Bool {
        print (#function)
        return true
    }

    func applicationDidBecomeActive(_ application: UIApplication)
{
        print (#function)
    }

    func application(_ application: UIApplication,
configurationForConnecting connectingSceneSession: UISceneSession,
options: UIScene.ConnectionOptions) -> UISceneConfiguration {

        print (#function)
        return UISceneConfiguration(name: "Default Configuration",
sessionRole: connectingSceneSession.role)
    }

    func application(_ application: UIApplication,
didDiscardSceneSessions sceneSessions: Set<UISceneSession>) {
        print (#function)
    }
```

Листинг 3.

Описание используемого SceneDelegate

```
     class SceneDelegate: UIResponder, UIWindowSceneDelegate {


         var window: UIWindow?
         func scene(_ scene: UIScene, willConnectTo session:
UISceneSession, options connectionOptions:
UIScene.ConnectionOptions) {
             print(#function)
             connectionStatus = checkConnection()
             guard let _ = (scene as? UIWindowScene) else { return
}
         }
         func sceneDidDisconnect(_ scene: UIScene) {
             print(#function)
         }
         func sceneDidBecomeActive(_ scene: UIScene) {
             print(#function)
         }
         func sceneWillResignActive(_ scene: UIScene) {
```

```
        print(#function)
    }
    func sceneWillEnterForeground(_ scene: UIScene) {
        print(#function)
    }
    func sceneDidEnterBackground(_ scene: UIScene) {
        print(#function)
    }
}
```

На рисунке 2 показано выполнения метода application (_ application: UIApplication, willFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey : Any]? = nil) -> Bool. Приложение только начало свою загрузку, подготавливая основную логику и методы к функционированию.

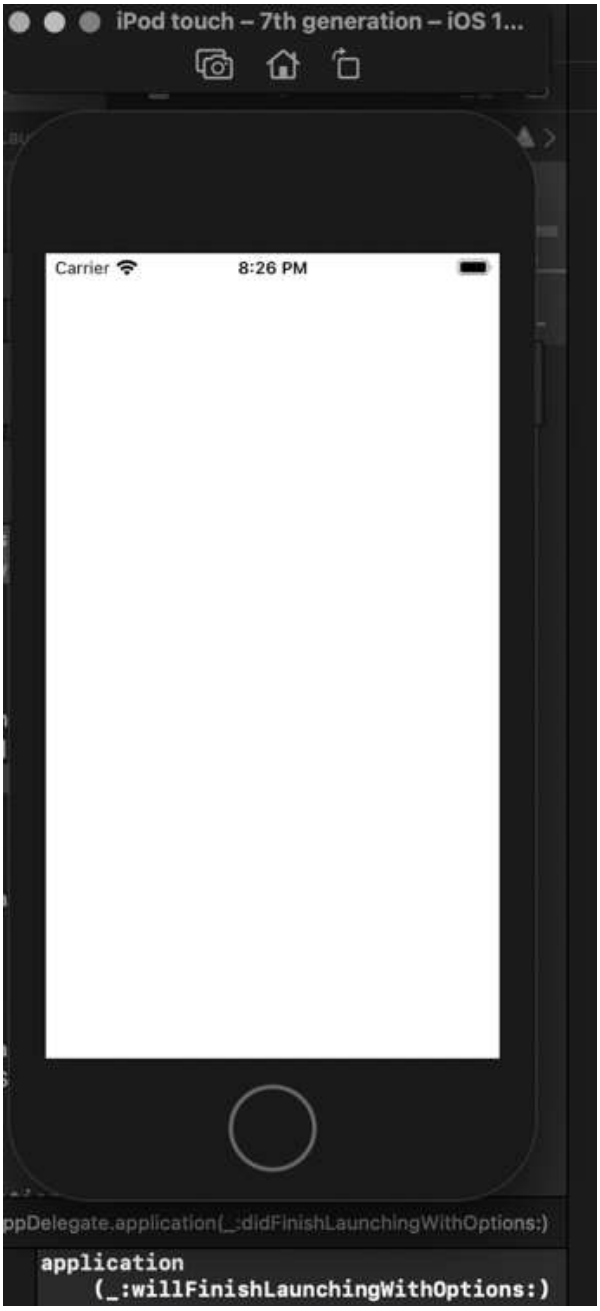



**Рисунок 2 – срабатывание метода application (_:willFinishLaunchingWithOptions:)**

На рисунке 3 показано срабатывание метода application (_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool. Приложение успешно загрузило основную логику и методы, а также зарезервировала память для функционирования.




**Рисунок 3 – срабатывание метода application (_:didFinishLaunchingWithOptions:)**

На рисунке 4 показано срабатывание метода scene(_ scene: UIScene, willConnectTo session: UISceneSession, options connectionOptions: UIScene.ConnectionOptions). Приложение начало загрузку и подключение окна для экранов графического интерфейса. Кроме этого, приложение также выполнило метод checkConnection(), результат выполнения которого будет показан позже.




**Рисунок 4 – срабатывание метода scene (_:willConnectTo:options:)**

На рисунке 5 показано срабатывание метода func sceneWillEnterForeground(_ scene: UIScene). Приложение подготавливает экран графического интерфейса для его отображения на экране.




**Рисунок 5 – срабатывание метода sceneWillEnterForeground(_:)**

На рисунке 6 показано срабатывание метода sceneDidEnterBackground(_ scene: UIScene). Приложение отобразило экран графического интерфейса на дисплее устройства. Как можно заметить, приложение не смогло установить подключение к серверу.




**Рисунок 6 – срабатывание метода sceneDidEnterForeground(_:)**

На рисунке 7 показано срабатывание метода sceneWillResignActive(_ scene: UIScene). Приложение получило сигнал от системы о том, что в скором времени окно и интерфейсы будут убраны с дисплея. В процессе выполнения данного метода приложение сохраняет основные параметры экрана графического интерфейса и основные данные, которые могут быть использованы при повторной загрузке приложения, которая будет начинаться не с этапа запуска приложения, а с этапа, на котором выполнение приложения было прервано.




**Рисунок 7 – срабатывание метода sceneWillResignActive(_:)**

На рисунке 8 показано срабатывание метода sceneDidResignActive(_ scene: UIScene). Окно и все соответствующие экраны были убраны с дисплея, однако само приложение осталось в памяти устройства и при запуске приложения вновь оно будет загружаться не с самого начала, а с момента, при котором оно было прервано.




**Рисунок 8 – срабатывание метода sceneDidResignActive(_:)**

## Выводы

В статье была проанализирована новая модель жизненного цикла iOS-приложения, используемого с iOS 13, были описаны основные этапы жизненного цикла и методы, применяемые на каждом этапе. На практическом примере было показано применение большинства из описанных выше методов. Данные методы не обязательно переопределять в каждом приложении, однако знание и умение их применять в необходимых ситуациях позволят более гибко реализовывать необходимый функционал.

## Использованные источники:

1. Protocol: UIApplicationDelegate. [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate (дата обращения: 19.02.2021)

2. Article: Managing Your App's Life Cycle. [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/app_and_environment/managing_your_app_s_life_cycle (дата обращения: 19.02.2021)
3. Instance Method: application(_:willFinishLaunchingWithOptions:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1623032-application (дата обращения: 20.02.2021)
4. Instance Method: application(_:didFinishLaunchingWithOptions:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1622921-application (дата обращения: 20.02.2021)
5. Instance Method: application(_:configurationForConnecting:options:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/3197905-application (дата обращения: 20.02.2021)
6. Instance Method: sceneWillEnterForeground(_:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197918-scenewillenterforeground (дата обращения: 21.02.2021)
7. Instance Method: sceneDidBecomeActive(_:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197915-scenedidbecomeactive (дата обращения: 21.02.2021)
8. Instance Method: sceneWillResignActive(_:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197919-scenewillresignactive (дата обращения: 22.02.2021)
9. Instance Method: sceneDidEnterBackground(_:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197917-scenedidenterbackground (дата обращения: 22.02.2021)
10. Instance Method: sceneDidDisconnect(_:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197916-scenediddisconnect (дата обращения: 22.02.2021)

11. Instance Method: applicationWillTerminate(_:). [Электронный ресурс]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1623111-applicationwillterminate (дата обращения: 22.02.2021)

**Sources used:**

1. Protocol: UIApplicationDelegate. [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate (accessed: 19.02.2021)
2. Article: Managing The Lifecycle Of Your Application. [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/app_and_environment/managing_your_app_s_life_cycle (accessed: 19.02.2021)
3. Instance method: application ( _ : willFinishLaunchingWithOptions:). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1623032-application (accessed: 20.02.2021)
4. Instance method: application ( _ : didFinishLaunchingWithOptions:). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1622921-application (accessed: 20.02.2021)
5. Instance method: application ( _ : configurationForConnecting: options:). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/3197905-application (accessed: 20.02.2021)
6. Instance method: sceneWillEnterForeground ( _ :). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197918-scenewillenterforeground (accessed: 21.02.2021)
7. Instance method: sceneDidBecomeActive ( _ :). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197915-scenedidbecomeactive (accessed: 21.02.2021)

8. Instance method: sceneWillResignActive ( _ :). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197919-scenewillresignactive (accessed: 22.02.2021)
9. Instance method: sceneDidEnterBackground ( _ :). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197917-scenedidenterbackground (accessed: 22.02.2021)
10. Instance method: sceneDidDisconnect ( _ :). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiscenedelegate/3197916-scenediddisconnect (accessed: 22.02.2021)
11. Instance method: applicationWillTerminate ( _ :). [Electronic resource]. URL: https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1623111-applicationwillterminate (accessed: 22.02.2021)