OTUS

help@otus.ru

Общая стоимость

50 000 руб.

В месяц: 12 500 руб.

iOS Разработчик

Вся сила и мощь Swift 5

Продолжительность

5 месяцев, 4 часа в неделю

Начало занятий

23 мая

Длительность курса: 144 академических часа

1 Эффективный UIKit и CoreGraphics

1	**Проектируем UI без кода и зачем это нужно, storyboard, xib**	– Решение организационных вопросов. – Настройка environment: Xcode, git, scripts – Возможности UIKit в создании интерфейсов без кода и зачем это нужно

2 **Внутренности UIKit, Preservation и Restoration техники**

— Структура классов UIKit
— Life cycle приложения
— UIResponder и UIViewController Life cycles
— ViewControllerLifecycleBehaviour
– Preservation и Restoration состояния

Домашние задания

1 Реализация каркаса/прототипа приложения с помощью Storyboard с поддержкой Restoration

---

3 **Изучаем внутренности и хитрости UIScrollView, продвинутое использование компонента**

– Взаимосвязь между contentSize, contentOffset, contentInset
– Реализация бесконечного скроллинга
– Использование Stationary views
– Кастомная обработка тачей
– zoomScale и redraw
– UIScrollView и Autolayout

---

4 **Отобораженние структурированных данных UICollectionView, UITableView, UIStackView**

– Использование Flow Layout
– Комбинирование UICollectionView с PageControl
– UITableView Sections
– Оптимизация рендеринга таблицы
– UIStackView и Autolayout
– Решение проблем использования UIStackView, Compression Resistance и Hugging

Домашние задания

1 Реализация меню на UICollectionView и UIPageControl

5	**SizeClasses, Anchors и анимирование Auto Layout**	– SizeClasses, использование UITraitCollection – Продвинутая адаптивность на iOS без костылей – Safe Area – Layout Anchors – Анимирование Constraints
6	**Создание сверхсложных интерфейсов с помощью CoreGraphis и CoreAnimation**	– Работа с CALayer, различия с UIKit – CGContext – Программное рисование с помощью UIBezierPath – CAAnimation и CAAnimationGroup Домашние задания 1 Создание анимированный диаграммы на CoreGraphics

2 Foundation без сторонних фреймвоков и Swift 5 Standard Library

1 **Необычная система типов Swift, структуры данных, Generics**

— Теория типов и Compound и Named типы
— Метатип и вложенные типы
— Protocol Composition
— Generics
— Создание кастомных структур данных

---

2 **Sequences и коллекции, асимптотический анализ: O(1), O(N), O(N*log(N)), O(n^2)**

— Sequence и IteratorProtocol
— Type-erased типы: AnySequence, AnyIterator, AnyCollection
— Lazy Wrappers
— Wrappers for Algorithms
— Асимпотический анализ встроенных и кастомных структур данных

Домашние задания

1 Реализация механизма тестирования алгоритмов на производительность и отображения результатов

---

3 **Использование всей мощи String, Literals vs. UnicodeScalar, UTF-16**

— Сравнение суффиксов и префиксов и другие способы сравнения строк
— Работа с utf8 и utf16 представлениями
— Ипользование подстрок и Ranges, StringProtocol
— Regex

---

4 **Региональные форматы и локализация iOS приложения**

— Парсинг и представление телефонных номеров
— Форматирование дат согласно региону и локали, POSIX спецификация
— Работа с единицами измерения и валютами
— Корректная локализация приложения на несколько языков и регионов

Домашние задания

1 Создание ячейки с поддержкой региональной локализации

---

5 **Ассоциативные типы, Type Erasure, «сахарные» типы данных, диспетчеризация вызовов в Swift 5**

— 3 типа диспатчеризации в Swift: direct, dynamic, message
— Associated Types
— PATs и динамическая диспатчеризация
— Другие способы реализации паттерна Type Erasure
— Как работают типы в SIL (Swift Intermediate Language)

---

6 **Компилятор LLVM, AST, создание собственных операторов**

— Как работает LLVM: SIL, IR
— Как некоторые типы представлены в SIL и для чего это нужно знать
— Особенности и хитрости компиляции
— Перезагрузка и создание операторов

Домашние задания

1 Проектирование и реализация своей собственной структуры данных и оценка ее эффективности

3 Современная архитектура мобильных приложений

1 **Современные паттерны проектирования, принцип SOLID и его целесообразное применение**

— SOLID и как получить от него пользу
— Другие принципы: KISS, DRY/DIE, YAGNI, BDUF, SOC
— Необходимые паттерны для сегодняшней мобильной разработки: Adapter, Memento, Observer, Strategy, Factory, Command, Composite, Iterator, Mediator, Proxy, Template Method, Singleton и где они приненимы на iOS

---

2 **MVP, MVCS, MVVM, архитектурные паттерны, модуляризация, Clean Architecture**

— MV(X) архитектурные паттерны
— Модуляризация приложения, способы: Frameworks, Cocoapods, JSCore, SPM
— Clean Architecture

Домашние задания

1 Создания каркаса для модуляризированного приложения

---

3 **Protocol Oriented Programming (POP)**

— Про Наследовании и ООП
— Миксины/трейты: Protocol Extensions
— Type Constraints
— Плюсы и минусы подхода

---

4 **Dependency Injection, SOA, слоистая архитектура**

— Inversion of Control паттерн
— ServiceLocator и инжектинг
— Разделение архитектуры на слои и что это дает

Домашние задания

1 Реализация Service Locator и Dependency Injection и изолированности слоев в приложении

---

5 **Связывание разных частей приложения Observing, Signals, Callbacks**

— Observing и broadcasting, нужен ли нам KVO
— Плюсы и минусы Delegation, виды делегатов
— Callbacks
— Signals ans Slots и причем здесь Rx

4 Многозадачность

1 **Проблемы многозадачности и способы их решения, GCD**

— Антипаттерны и проблемы: Priority Inversion, Race condition, Deadlock, Resource contention, Starvation, Non-deterministic and Fairness
— Использование GCD: QoS, Queues, Main Queue и Main Thread

Домашние задания

1. Реализация асинхронного выполнения задач и оценка эффективности подхода

---

2 **Внутренности GCD(libdispatch), OperationQueue**

— Плюсы и минусы OperationQueue
— Внутренности libdispatch: пул тредов, continuation, QoS и как зная это лучше использовать очереди

---

3 **RunLoop & POSIX Threads, Инструменты синхронизации, Lock, Mutex**

— RunLoop и чем сегодня он может нам быть полезен
— pthreads
— Виды локов: NSLock, NSRecursiveLock, Spinlock, Mutex, Semaphore
— Dispatch Barriers
— Trampoline техника

Домашние задания

1. Реализация общей конкурентной очереди на приложение, создание внутренней системы тасков

5 Networking и хранение данных

1 **Новый Network-фреймвок, URLSession, Codable**

— Network феймвок, HTTP, REST, Sockets, GraphQL
— URLSession
— Сериализация и десериализация с помощью Codable

---

2 **Когда использовать Files, чистый SQLite, способы кеширования**

— Виды кеширования
— SQLite и другие DB* альтернативы
— NoSQL
— Files и File System

Домашние задания

1 Имплементация сериализации и сохранения данных на backend

---

3 **CoreData — основные стратегии использования**

— NSManagedObject, NSPersistentStoreCoordinator, NSManagedObjectContext
— NSPersistentContainer
— Модель данных
— CRUD на Core Data

---

4 **Realm**

— Плюсы и минусы Realm
— Модель данных на Realm
— Realm Browser
— CRUD на Realm

Домашние задания

1 Реализация поддержки оффлайн режима в приложении

6 Целесообразный Rx

1 **Отличие императивного и реактивного подхода, концепция Observable**

— Чем реактивный подход может быть лучше
— Observable Sequences
— RxSwift 4.x
— Marble Diagrams

---

2 **Promises, Signals и как это в Rx. Функциональное программирование**

— Rx подход без RxSwift: PromiseKit, Signals
— Идея распространение изменений
— RxSwift Transformations: Map, FlatMap, Scan, Buffer

Домашние задания

1 Реализация каркаса приложения используя декларативный подход и Rx

---

3 **Основные паттерны: Composition, Aggregation, Cancellatiion**

— Rx паттерны
— Фильтрация: Filter, DistinctUntilChanged
— Комбинирование: StartWith, Merge, Zip

---

4 **UI Patterns: Driver & Action, быстрое создание полноценного UI на Rx**

— MVVM и RxSwift
— RxCocoa
— Создание UI на Rx

Домашние задания

1 Создание UI слоя приложения на Rx

7 Организация разработки

1 **Тестирование кода XCTest, UITest, fastlane и CI**

— Test-Driven Development (TDD)
— Теория тестирования
— Использование XCTest
— Зачем нужен UITest
— Сборка CI (Continuous Integration) на fastlane

---

2 **Git-flow, автоматизизация workflow**

— Продвинутое использование git
— Фича, хотфикс, релизный цикл, master ветка
— Использование комманд git-flow

Домашние задания

1 Покрытие приложения юнит-тестами

---

3 **Как правильно написать резюме и развивать hard-skills**

— Почему важно корректно писать резюме
— Как выбирать работодателя чтобы развивать свой hard-skills
— Какие бывают работодатели
— Какие скрытые критерии отбора используются

Домашние задания

1 Написание работающего резюме

8 Проектная работа

1 **Написание приложения с нуля**

— Выбор темы для приложения
— Как генерировать идеи для простых приложений на основе известных «болей» пользователей
— Как использовать iOS платформу для генерации идей для приложений
— Подбор инструментов, помощь с стартом написания приложения

Домашние задания

1 Написание приложения с нуля