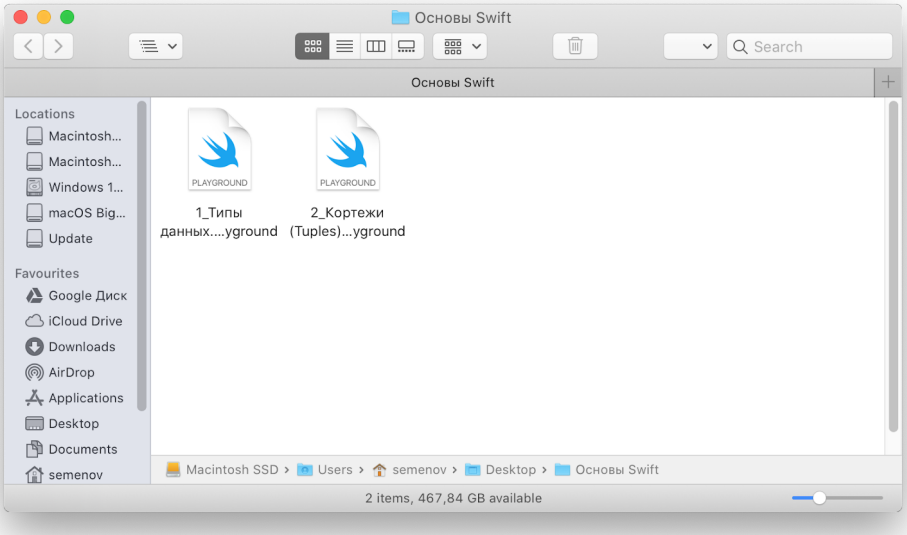
План обучения до уровня junior iOS-разработчик

Основной учебник: Усов Василий “Swift. Основы разработки приложений под iOS, iPadOS и macOS.” 5-е изд., дополненное и переработанное. - СПб.: Питер, 2020.

1. Формируем базовые знания синтаксиса и понимание возможностей языка
    a. **Задача:** изучаем главы I-IV
    b. **Ссылки:** https://books.apple.com/us/book/swift-programming-language/id881256329 - учебник от Apple
    https://docs.swift.org/swift-book/index.html - вспомогательный учебник
    c. **Примечание:** обязательная реализация примеров кода из учебника, а не просто чтение. Заводим папку для обучения, где создаем по каждой теме playground файл с реализацией. Впоследствии эти файлы будут шпаргалкой, если что-то забылось



2. Закрепляем знания на практике
    a. **Задача:** убедиться, что мы усвоили материал и можем двигаться дальше
    b. **Ссылка:** https://swiftme.ru/oglavlenie-kursa-1175/ - практические задания
    c. **Примечание:** если после изучения темы остались вопросы, не ленимся вернуться и перечитать. Если не удалось разобраться до конца, задать вопросы мне в личку

3. Знакомимся с интерфейсом Xcode
    a. **Задача:** изучаем главу V
    b. **Примечание:** научиться взаимодействовать с IDE Xcode для создания проектов, просмотра справки и запуска программ

4. Изучаем принципы Объектно-Ориентированного Программирования (ООП)
    a. **Задача:** изучаем главу VI
    b. **Примечание:** это фундамент для последующей разработки. Нужно отнестись к этому разделу со всей серьезностью

5. ООП, ПОП (Протоколо-Ориентированное Программирование) и ФОП (Функционально-Ориентированное Программирование)
    a. **Задача:** посмотреть на принципы с разных сторон для полноты картины и закрепления
    b. **Ссылка:** Protocol-Oriented Programming in Swift - WWDC 2015 - Videos - ПОП
    https://medium.com/bitmountn/oop-vs-pop-vs-fop-c20430e6e620 - различия
    c. **Примечание:** при детальном разборе принципа ООП обращаем внимание не только на трех китов (наследование, полиморфизм, инкапсуляция), но также и на принцип “Абстракция”

6. Императивное и Декларативное программирование
    a. **Задача:** различать парадигмы разработки
    b. **Ссылки:** https://habr.com/ru/post/324688/ - отличия главных подходов
    https://ru.wikipedia.org/wiki/Парадигма_программирования - все подходы, если интересно для расширения кругозора
    c. **Примечание:** главное уловить суть, в первом случае мы описываем КАК нужно получить желаемое, во втором ЧТО мы ходит получить

7. Reference type vs. Value type + Принцип “Copy On Write”
    a. **Задача:** понимать, как происходит работа с памятью внутри приложения
    b. **Ссылки:**
    https://medium.com/@abhimuralidharan/difference-between-value-type-and-a-reference-type-in-ios-swift-18cb5145ad7a - различия типов
    https://developer.apple.com/swift/blog/?id=10 - о различиях в доках Apple
    https://www.hackingwithswift.com/example-code/language/what-is-copy-on-write - принцип “copy on write” передачи данных по значению
    https://marcosantadev.com/copy-write-swift-value-types/ - также про принцип “copy on write”
    c. **Примечание:** какие типы данных как передаются внутри приложения. Понятие мутабельности

8. Закрепить материал последних разделов VI главы
    a. **Задача:** научиться писать гибкий и безопасный код
    b. **Ссылки:**
    https://www.uraimo.com/2016/10/27/unowned-or-weak-lifetime-and-performance/ - различия weak/unowned
    https://medium.com/@hhadevs/strong-unowned-weak-в-чем-разница-b293963f3375 - различия на русском
    https://habr.com/ru/post/444336/ - также по русски написано про различия
    https://marcosantadev.com/swift-arrays-holding-elements-weak-references/ - массив слабых ссылок
    https://medium.com/flawless-app-stories/you-dont-always-need-weak-self-a778bec505ef - почему не нужно всегда использовать weak
    https://www.swiftbysundell.com/articles/different-flavors-of-type-erasure-in-swift/ - дженерики
    c. **Примечание:** чаще всего приложения падают из за неправильной работы с памятью, а дублирование функционала порождает много кода, который тяжело поддерживать

9. Захват ссылок + escaping/non-escaping замыкания
    a. **Задача:** понимать принципы управления памятью в случае отложенных операций
    b. **Ссылка:**
    https://medium.com/swiftcommmunity/what-do-mean-escaping-and-nonescaping-closures-in-swift-d404d721f39d - различия и примеры использования

10. Знакомимся с разработкой под iOS
    a. **Задача:** прокачиваем использование Xcode, учимся основам создания приложений под платформу iOS в главе VII, раздел 38
    b. **Примечание:** нужно разобраться с тем, что такое storyboard и как с его помощью верстать экраны. Учимся связывать визуальное представление нашего приложения в storyboard с кодом

11. Архитектура построения приложения и паттерны
    a. **Задача:** изучить раздел 39 главы VII
    b. **Примечание:** изучаем главную архитектуру MVC, предлагаемую и используемую разработчиками Apple. Познаем основные паттерны проектирования нашего кода