**Лабораторная работа №1**

**Разработка мобильных приложений на Swift. Введение.**

**Начало работы с Xcode 9.**

*Следует предварительно ознакомиться с приведенным здесь материалом (до лабораторной работы изучить теорию). Лабораторная работа включает теорию и задания для выполнения.*

**Введение в swift**

Новый язык программирования Swift является уникальным инструментом для разработки мобильных приложений для iOS и OS X.

Swift: удачная интеграция современных языков

Apple начала работать над созданием Swift в 2010 году, используя платформы нескольких языков. В итоге Swift получил модернизированные фреймворки от Cocoa и Foundation, которые без особого труда могут заимствовать возможности Swift в процессе их взаимодействия. Был значительно упрощен процесс, отвечающий за управление памятью с ARC (автоматический подсчет ссылок).

Итак Swift — мультипарадигменный объектно-ориентированный язык программирования, созданный компанией Apple для разработчиков iOS и OSX. Swift работает с фреймворками Cocoa и Cocoa Touch и совместим с основной кодовой базой Apple, написанной на Objective-C. Swift задумывался как более безопасный язык в сравнении с Objective-C. Язык поддерживается в среде программирования Xcode; программы на нем компилируются при помощи Apple LLVM и используют рантайм Objective-C, что делает возможным использование обоих языков (а также pure С и С++) в рамках одной программы.

Разработка Swift началась в 2010 году Крисом Латтнером, руководителем отдела разработки инструментов разработки в Apple и одним из основных разработчиков LLVM.

Не нужно импортировать библиотеку для работы с вводом/выводом или работой со строками. Код, написанный в глобальной области видимости (global scope), используется как входная точка для приложения, поэтому функция main() не нужна. Также не нужно ставить точки с запятой в конце каждого оператора.

Особенность Swift состоит в поддержке инновационной функции playground. Она делает возможным проводить любые эксперименты, видеть моментально их результат, не требуя запуска приложения.

Что еще?

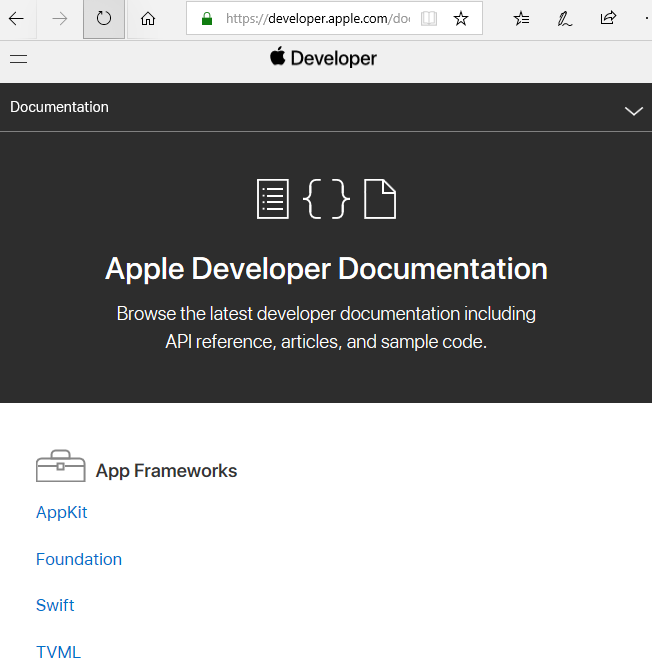
1. Не нужно импортировать библиотеку для работы с вводом/выводом или работой со строками.
2. Код, написанный в глобальной области видимости (global scope), используется как входная точка для приложения, поэтому функция main() не нужна.
3. Также не нужно ставить точки с запятой в конце каждого оператора.

**Задание 1.1 Получение учетной записи Apple ID регистрация в центре Apple-разработчиков .**

Для этого необходимо пройти по ссылке <https://developer.apple.com/register/>

Apple ID — это учетная запись, которая позволяет получить доступ к сервисам, предоставляемым фирмой Apple.

На стартовой странице Центра разработчиков необходимо перейти по ссылке Developer, расположенной в верхней части страницы, после чего откроется страница, содержащая ссылки на основные ресурсы для разработчиков.



В Центре разработчиков вы можете получить доступ к огромному количеству различной документации, видео, примеров кода — ко всему, что поможет создавать отличные приложения . Регистрация в качестве разработчика бесплатна

С недавних пор у программистов появилась возможность попрактиковаться в разработке на Swift под ОС Windows . Для программирования под Windows существует пакет Swift for Windows, который устанавливает необходимое окружение . Последнюю версию вы можете найти на портале http://swiftforwindows.github.com.

Все последующие примеры необходимо будет проверять и вводить в IDE XCode - <https://developer.apple.com/download/> (он уже у вас установлен в лаборатории).

Далее надо завести себе Apple ID на <https://www.apple.com/> - но в лаборатории он тоже установлен.

**Введение в Хcode**

Изучение программирования на языке Swift мы начнем со среды разработки Xcode . Xcode распространяется на бесплатной основе .

В Xcode интегрированы пакет iOS SDK, редактор кода, редактор интерфейса, отладчик и многое другое. Также в него встроены симуляторы iPhone, iPad, Apple Watch и Apple TV . Это значит, что все создаваемые приложения вы сможете тестировать прямо в Xcode (без необходимости загрузки программ на реальные устройства).

В нем содержится:

1) UIKit — фреймворк для построения пользовательского интерфейса. Он содержит в себе различные элементы управления (кнопки, текстовые поля, панель навигации и другие).

2) Cocoa Touch — это коллекция библиотек Apple, которая содержит множество функциональных фреймворков iOS, например UIKit для построения пользовательского интерфейса, SpriteKit для создания 2D-игр, MapKit для работы с картами, Core Graphics для рисования, Core Animation для анимации объектов и т. д.

3) NeXTSTEP — ОС компании NeXT.

4) iOS Simulator — инструмент, который входит в состав среды Xcode. С его помощью можно имитировать запуск приложений на виртуальных копиях iPhone и iPad.

5) Playgrounds — миниатюрные тестовые программы Swift, в рабочей области которых можно вводить код и моментально видеть результаты его выполнения. Они не используются в рабочих продуктах, но подходят для обучения.

Все ваши проекты начинают аккумулироваться справа в серой области (см. рисунок).

Есть три варианта попасть в Xcode. Сейчас нас будет интересовать *первый вариант* —использовать playground. Playground дает вам возможность писать код без создания приложения. Вы можете вызывать любые APIs, размещать любые вещи на экране, рисовать и т.д. Это своего рода «игровая площадка» для iOSпрограммирования.

Более подробное изучение состава и возможностей данной IDE отложим до момента изучения процесса разработки приложений .

Сейчас при изучении Swift вы будете взаимодействовать с рабочим интерфейсом playground-проектов .

**Playground**

В переводе на русский «песочница», представляет из себя окружение для того, чтобы изучить Swift. Выглядит это так: вы вводите строку кода и моментально видите результат его выполнения. После того, как вы убедитесь, что написанный вами код дает нужный результат, вы можете просто взять и перенести его в свой проект.

Например, при помощи Playground вы можете решить следующие задачи:

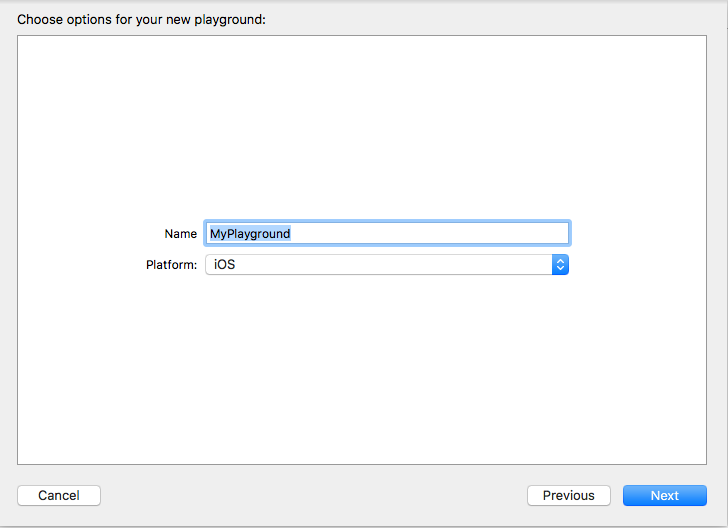
•освоить синтаксис Swift

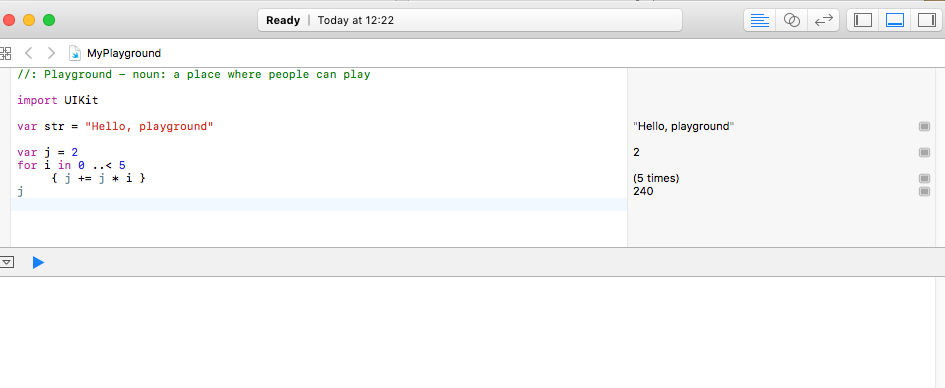
•поэкспериментировать с новыми API

•разработать новый алгоритм и понаблюдать за каждым шагом его выполнения

•провести простые арифметические вычисления

**Задание 1.2 Работа в**  **Playground**



Придумайте имя для своего Playground, нажмите кнопку Next и сохраните его куда-нибудь. Наберите небольшой код и посмотрите результаты.

Правый столбец называется панелью результатов (results sidebar) и для каждой строки в нем показано значение, которое будет получено в результате ее выполнения. Например, для строки 10 в столбце результата будет значение этой переменной.

Обратите внимание на строку 8, в ней панель результатов содержит не результат выражения, а то, сколько раз это выражение было посчитано (5 раз). Мы можем увидеть, какие значения были получены для этой строки. Чтобы это сделать, надо нажать на маленький прямоугольник рядом с текстом (5 times). После чего Playground станет выглядеть так:

Теперь внизу Playground отображается панель Timeline. Еще ее можно открыть, выбрав в меню пункт View->Assistant Editor->Show Assistant Editor.

Timeline позволяет увидеть, как менялось значение выбранного выражение во времени. Чтобы это сделать, используйте слайдер внизу. Попробуйте поперетаскивать его.

Кроме того, в Timeline показывает «Console Output». Это тот текст, который бы вывела ваша программа в консоль устройства или терминал.

Взгляните в верхний правый угол рабочего окна (выделенный красным овалом): там расположены шесть кнопок, позволяющие настраивать рабочую зону Xcode. Кнопки разбиты на две панели, по три кнопки в каждой . Левая панель отвечает за внешний вид рабочей зоны вашего окна и содержит следующие кнопки:

* Standart Editor Активна по умолчанию . Позволяет отобразить рабочую область в привычном нам виде: с редактором кода и областью результатов .
* Assistant Editor Добавляет дополнительную область «Ассистент разработки» . По умолчанию она расположена правее редактора кода и содержит различную вспомогательную информацию, включая графическое отображение playground-страниц
* Version Editor Позволяет работать с версионностью файлов . Данный режим работы доступен только при активной системе контроля версий . Правая панель позволяет отобразить ряд вспомогательных областей:
* Navigator Отображает ProjectNavigator, содержащий структуру вашего проекта .
* Debug Area Открывает уже знакомую вам DebugArea .
* Utilities Открывает панель Utilities, отображающую некоторые характеристики проекта, а также справочную информацию по выделенному элементу редактора кода .

Разберитесь во всеми режимами.