**Вариант 16**

2.2. Дополните приложение на Objective-C из 2.1 следующим функционалом, при выборе города выводит изобаржение университета города, и количество факультетов и студентов в нем. В приложении должны быть не менее двух городов из каждой страны и не менее двух стран, одна из которых Беларусь и университет — БГУ.

4.Разработать приложение «калькулятор»: преобразование из одной системы счисления в другую (2с\с, 8 с\с , 16с\с). В разных направлениях. Вывод информации на разных фонах

**ЗАДАНИЕ 1. ПРИЛОЖЕНИЕ ДЛЯ УПРАВЛЕНИЯ ФОНОМ НА ЯЗЫКЕ SWIFT**

**Цель:**

Создать iOS приложение с использованием объекта “Switch” для изменения фона контроллера. В положении “Вкл” – изображение “bg1.jpg”, в положении “Выкл” – изображение “bg2.jpg”. Изображения “bg1.jpg” и “bg2.jpg” ниже, слева направо соответственно (рис.8). Научиться работать с графическими ресурсами приложения.

В руководстве приведён пример исходного кода для Swft 3. Адаптировать для Swift 5 и, при необходимости, исправить ошибки, связанные с версией языка. В проект можно добавить любые изображения, но перед загрузкой в проект переименовать в bg1.jpg и bg2.jpg.



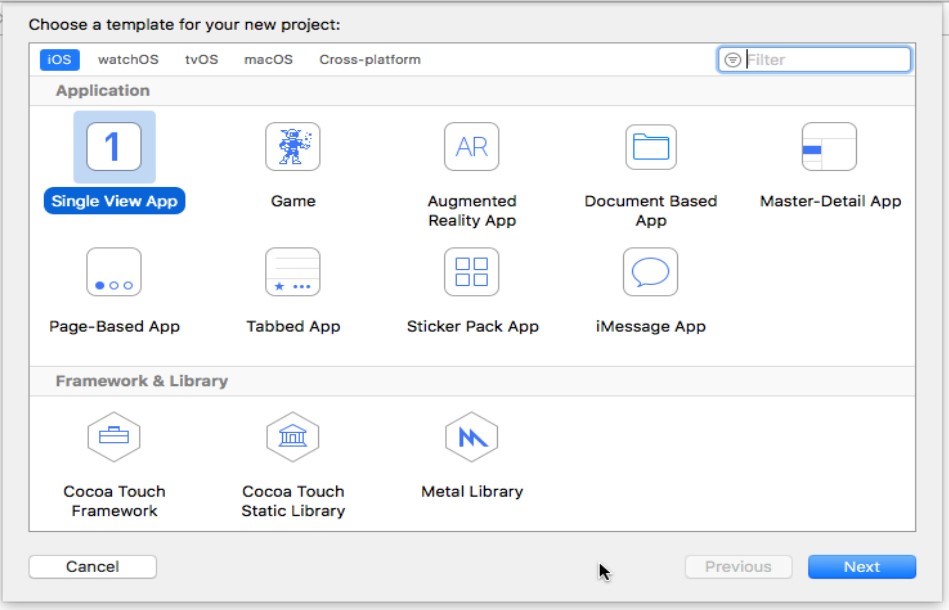
Рис.8. Изображения “bg1.jpg” и “bg2.jpg”

**Ход работы:**

Создание проекта Xcode.

Создаем проект Xcode, аналогично предыдущим лабораторным работам. Убираем галочку Use Core Data, Unit tests, UI Tests и выбираем iOS – Application – Single View Application.

1. Для начала давайте создадим проект: запустите Xcode и нажмите комбинацию клавиш CMD+Shift+N или выберите в меню File->New->Project



1. Теперь выберите Single View Application (App для Xcode 13 и выше) и нажмите Next.
2. На этом этапе надо придумать имя для приложения (Product Name), нажать Next и выбрать папку для сохранения. Например, параметры проекта для Xcode 13:

**Разработка интерфейса приложения**.

Откройте библиотеку объектов, , которая для Xcode 12 доступна на панели инспектора, а для Xcode 13 и выше в виде отдельного приложения (см. рис. Ниже). Находим в библиотеке объектов (Object library) переключатель Switch и перетаскиваем его в окно нашего View контроллера в файле Main.storyboard.

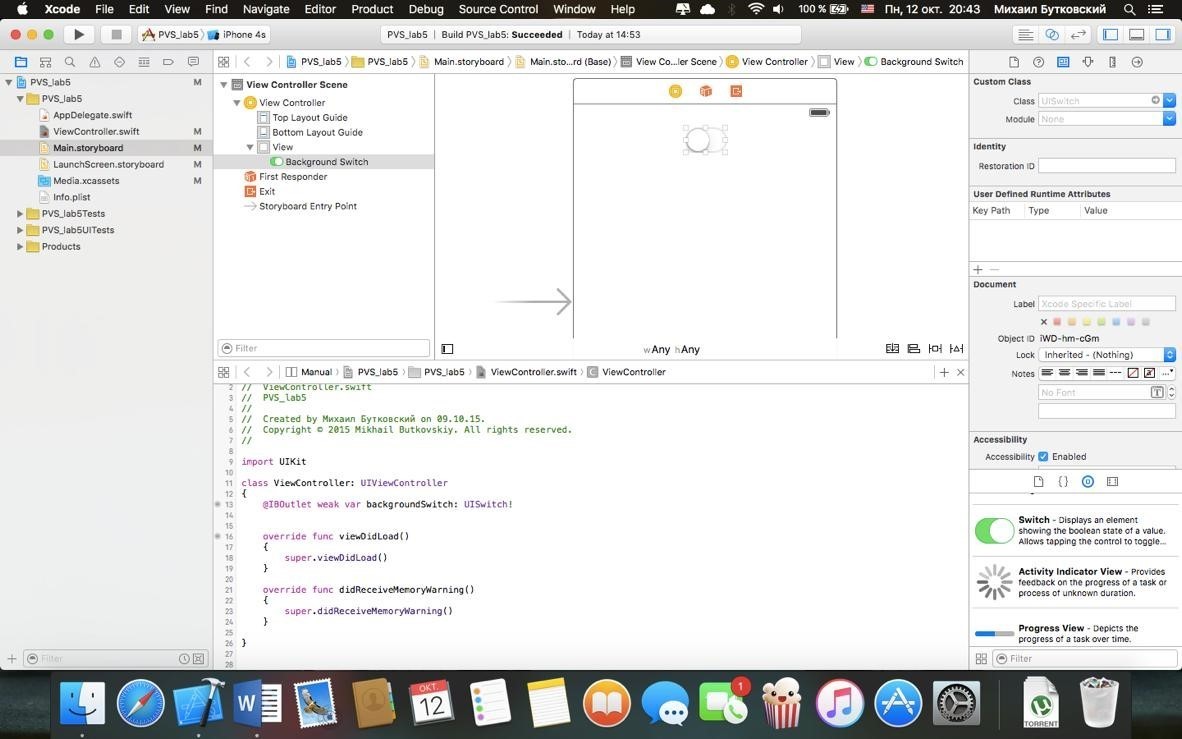
Располагаем его на экране как нам удобно. Также, добавляем объект Label, который будет индикатором переключателя Switch и располагаем его над переключателем.

Теперь необходимо связать объект Switch и Label с кодом нашей программы. Для этого открываем Assistant Editor. В одном окне выбираем файл ViewController.swift, а в другом – Main.storyboard. С зажатой клавишей ctrl перетаскиваем объект Switch в описание класса ViewController. Отпускаем ЛКМ и задаем имя объекту: например, backgroundSwitch и нажимаем Connect. Должна появится следующая строчка кода (рис. 9):

@IBOutlet weak var backgroundSwitch: UISwitch!

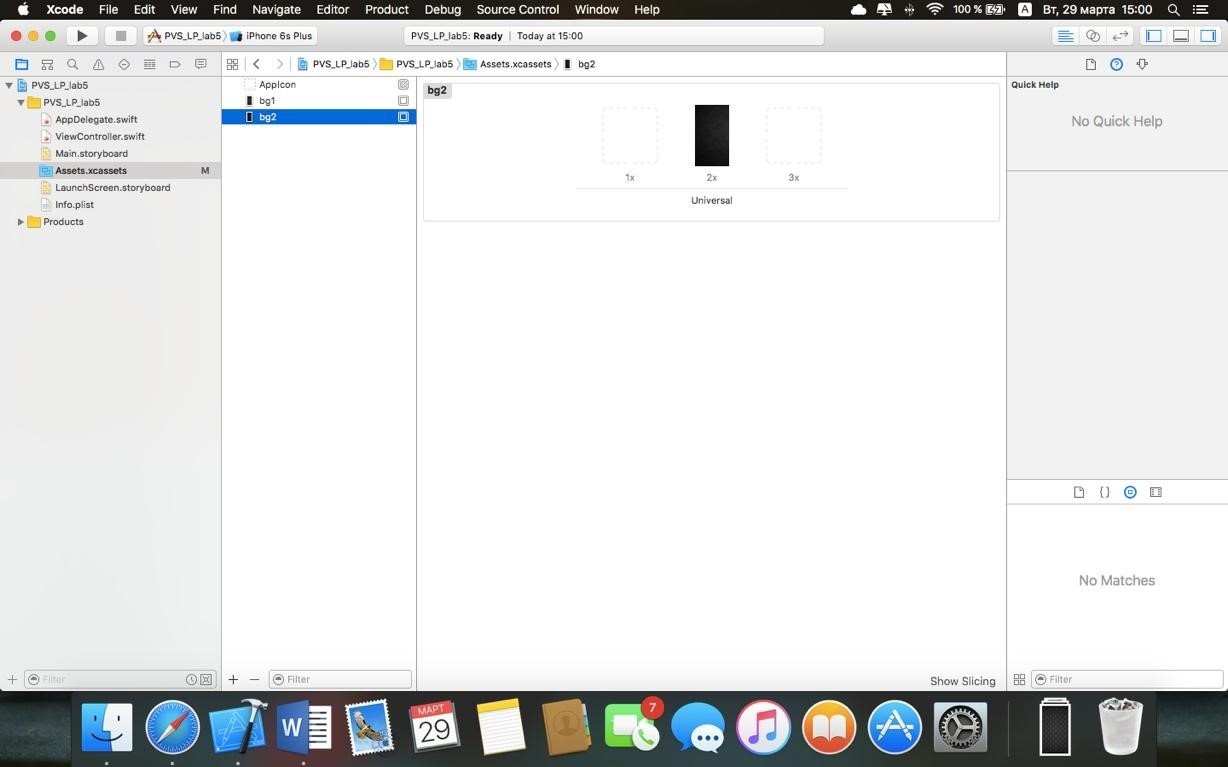
Повторяем эти действия для объекта Label:

@IBOutlet weak var switchIndicator: UILabel!



**Добавление изображений в проект Xcode.**

Для того, чтобы наше приложение использовало изображения, необходимо добавить их в специальное хранилище в нашем проекте. Для этого заранее изменяем разрешение изображений под разрешение iOS устройства, для которого вы создаете приложение (например – iPhone 5s, 640x1136 пикс.). Далее, в окне навигации открываем файл Assets.xcassets и добавляем набор изображений. Для этого нажимаем “+” (Add a group or image set) внизу списка и нажимаем “New Image Set”. Появляется окно с разными масштабами, под разные устройства (рис.10). Нам необходимо перетащить наше изображение в окошко 2x. Меняем название набора изображений с “Image” на “bg1”. Аналогично создаем набор изображений для “bg2”.



**Описание событий объекта “Switch” в коде, сборка и запуск приложения.**

Осталось описать события, которые будут происходить при изменении положения выключателя. С зажатой клавишей ctrl добавляем соединение для объекта “Switch” чуть ниже, чем предыдущая связь. Выбираем тип соединения Action и задаем имя, например: “backgroundSwitchTapped” и нажимаем Connect. У нас в коде появилась функция, в которой можно описывать события для объекта “Switch”.

В теле функции опишем следующее условие:

@IBAction func backgroundSwitchTapped(sender: AnyObject)

{

if backgroundSwitch.on

{

switchIndicator.text = "Background image: bg1.jpg"

view.backgroundColor = UIColor(patternImage: UIImage(named: "bg1")!)

} else

{

switchIndicator.text = "Background image: bg2.jpg"

view.backgroundColor = UIColor(patternImage: UIImage(named: "bg2")!)

}

}

Добавляем в функцию ViewDidLoad начальное положение объекта Switch и начальный статус OFF индикатора Label.

override func viewDidLoad()

{

super.viewDidLoad()

switchIndicator.textColor = UIColor.whiteColor()

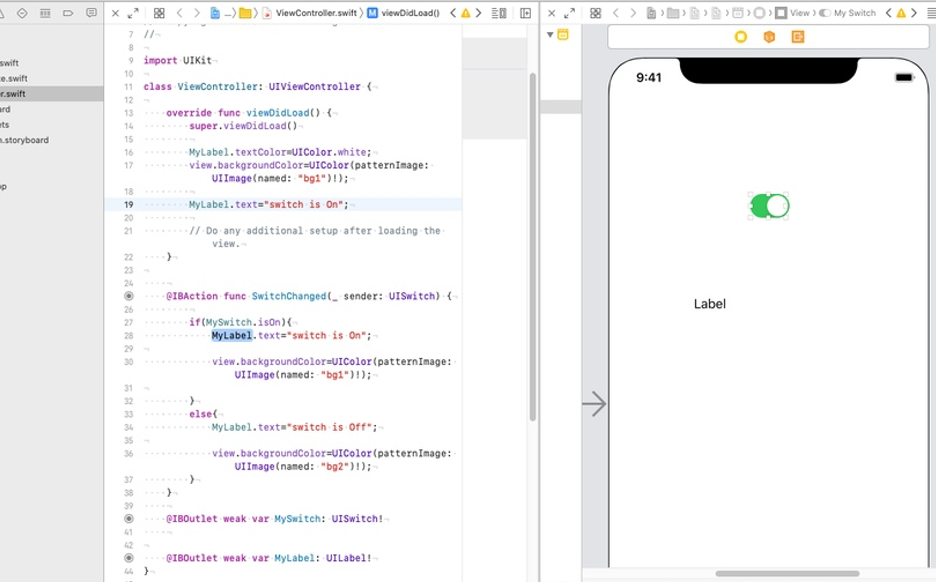
switchIndicator.text = "Background image: bg2.jpg"

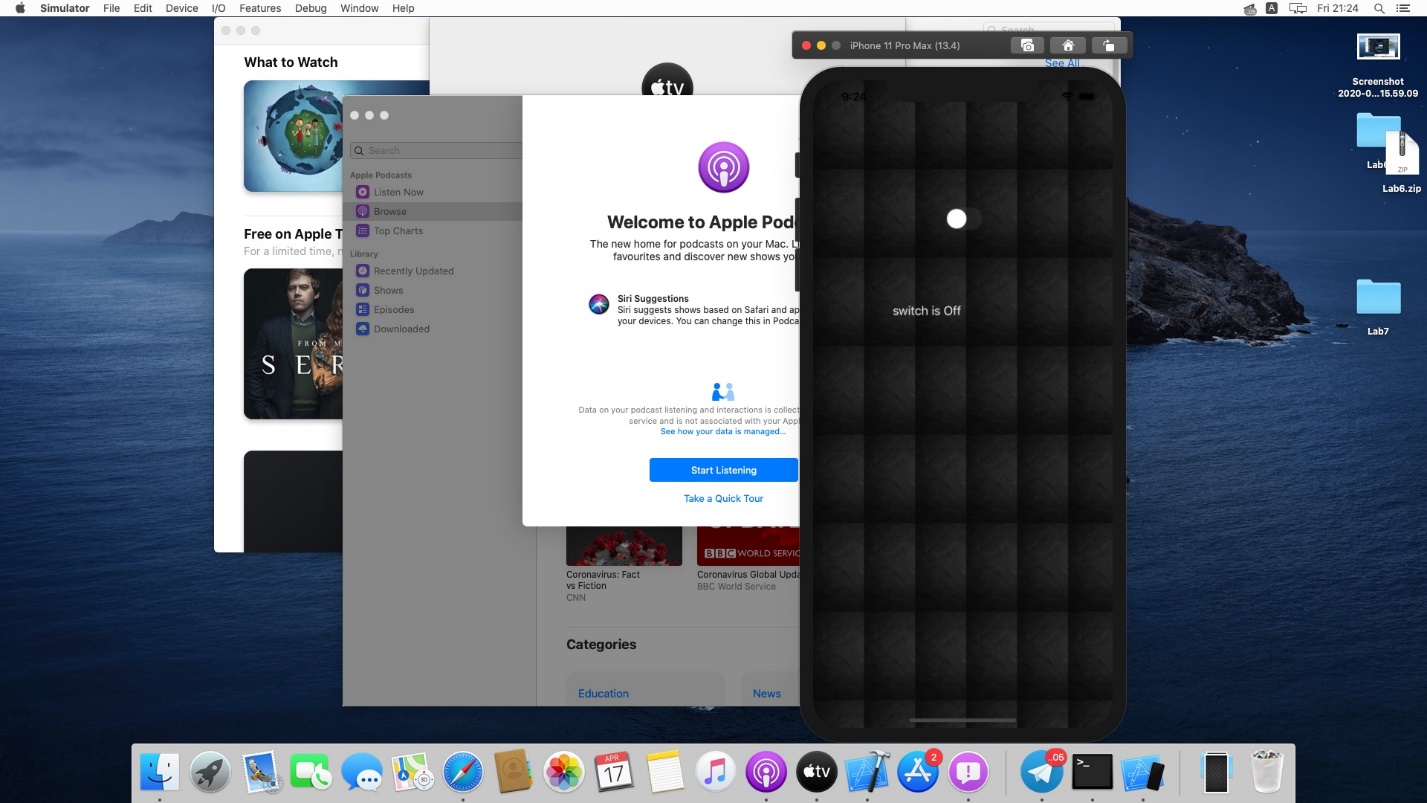
view.backgroundColor = UIColor(patternImage: UIImage(named: "bg2")!)

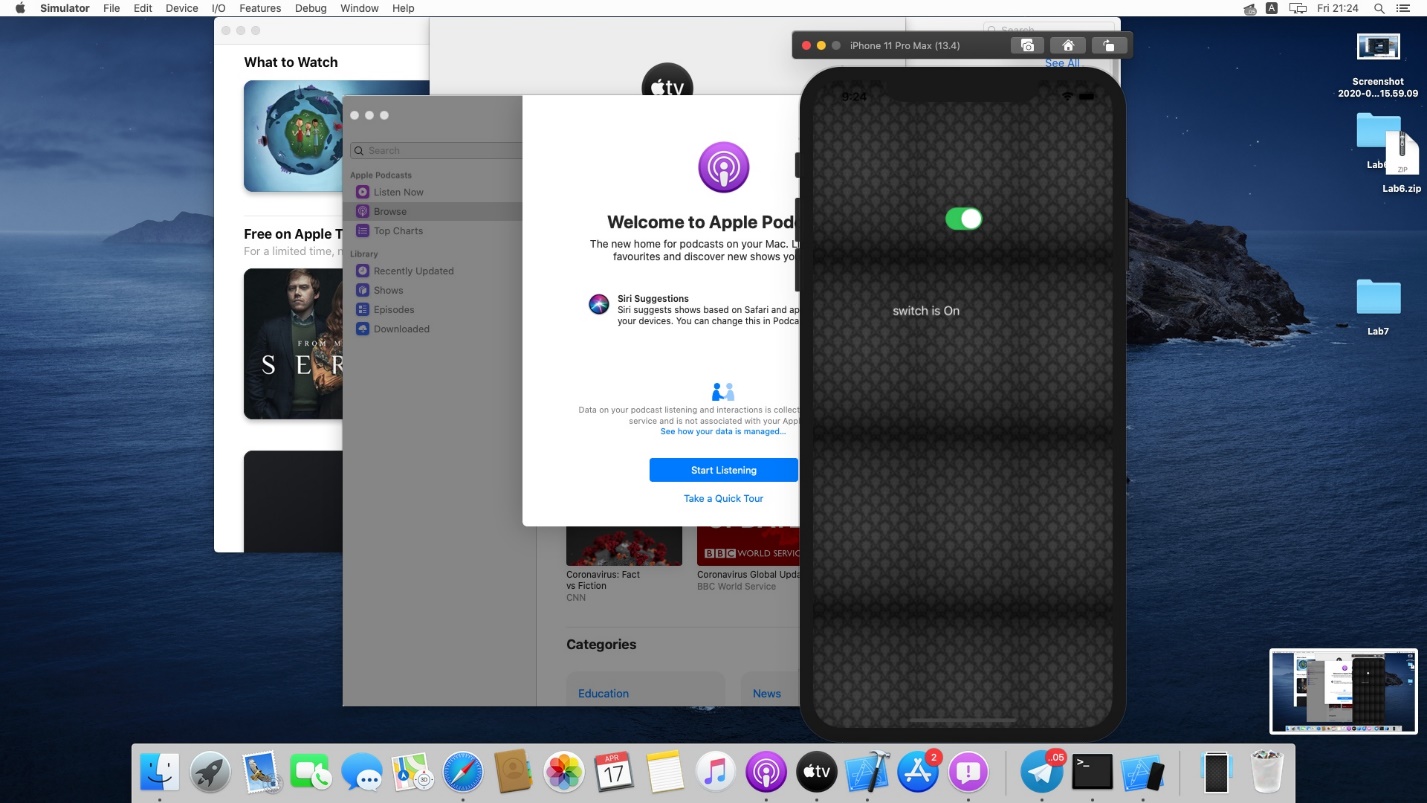
}

**Сборка проекта и запуск приложения.**

Выполняем команду cmd+R и проверяем работу приложения.







**ЗАДАНИЕ 2. ПОКАЗ ПОГОДЫ И ДРУГИХ СВЕДЕНИЙ О ГОРОДЕ НА ЯЗЫКЕ OBJECTIVE-C**

**УПРАЖНЕНИЕ 2.1**

Необходимо реализовать программу показа погоды для 6 городов. Цвет показателей температуры должен соответствовать погоде. Для хранения температуры и города использовать массив и/или словарь.

**Входные данные**

Строка, содержащая название города.

**Выходные данные**

Строка, содержащая информацию о температуре в городе и имеющая атрибут цвет, зависящий от температуры.

**УПРАЖНЕНИЕ 2.2.**

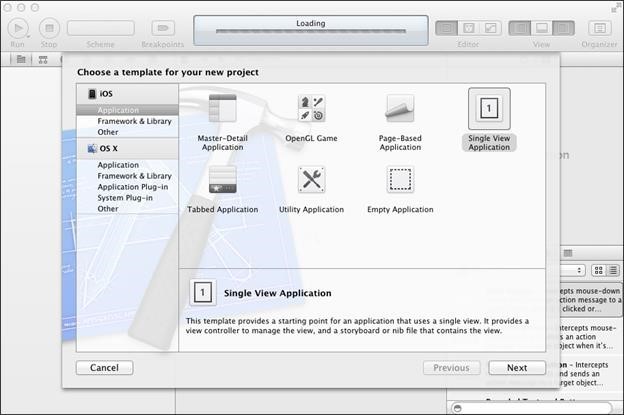
Внести изменения в спроектированное приложение согласно варианту. Исходный код приложений должен быть оформлен согласно руководству стиля для языка Objective-C и содержать комментарии к основным блокам кода:

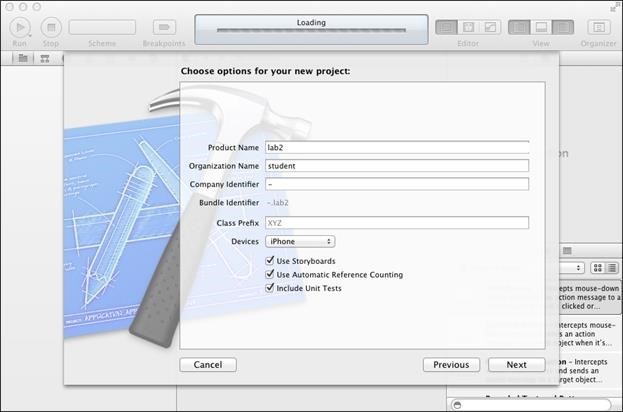
a) <https://github.com/DigDes/objective-c-style-guide>

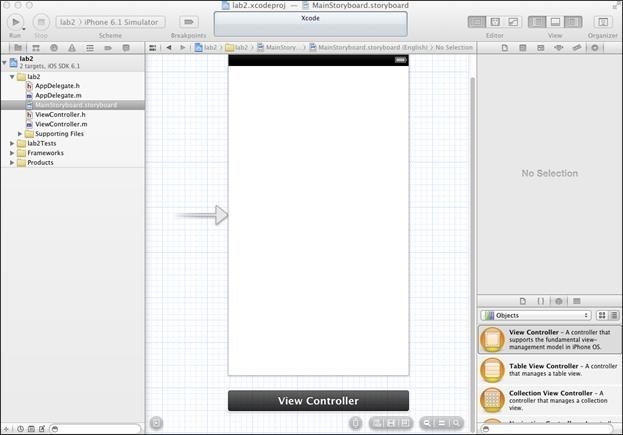
b) <https://github.com/raywenderlich/objective-c-style-guide>

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

Создайте новый проект вида Application → Single View Application (App в случае Xcode 13 и выше) (Рис. 11). Затем введите название проекта, имя разработчика, название организации (Рис. 12). Устройством выберите iPhone. Отметьте Use Storyboards.







Выберите папку и сохраните туда проект. В области навигации выберите MainStoryboard.storyboard. Окно редактора будет выглядеть примерно как на (Рис. 13).

На панели компонентов найдите компонент Label. Перетащите его на изображение пустого окна приложения iPhone (Рис. 14). Таким же образом добавьте кнопку (Round Rect Button).

Чтобы иметь возможность менять текст надписи в контроллере, необходимо добавить в него свойство надпись и привязать элементу интерфейса надписи (создать IBOutlet). Для этого выберите Assistant Editor верхней панели инструментов с названием Editor (иконка с изображением рубашки с бабочкой) (Рис. 15).

Ключевое слово IBOutlet (Interface Builder Outlets) предназначено для связи элемента интерфейса на форме созданным дизайнером форм и кодом.

Удерживая Ctrl перетащите надпись в файл ViewController.m между @interface и @end.



В результате в часть интерфейса контроллера добавится объявление IBOutlet (Листинг 1) . @property (weak, nonatomic) IBOutlet UILabel \*indicator;

**Листинг 1. Объявление IBOutlet**

Чтобы привязать к кнопке действие необходимо также с помощью Ctrl перетащить кнопку в область реализации между @implementation @end

(Создать IBAction) (Рис. 17). Введите название метода и выберите его тип. (Рис. 18)

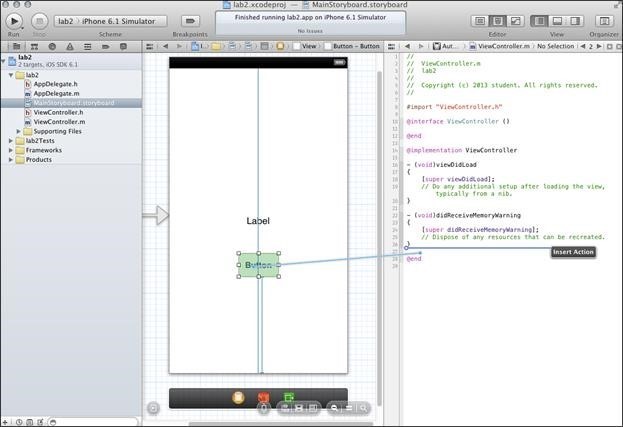
В результате, в часть реализации контроллера добавится код обработчика (Листинг 2):

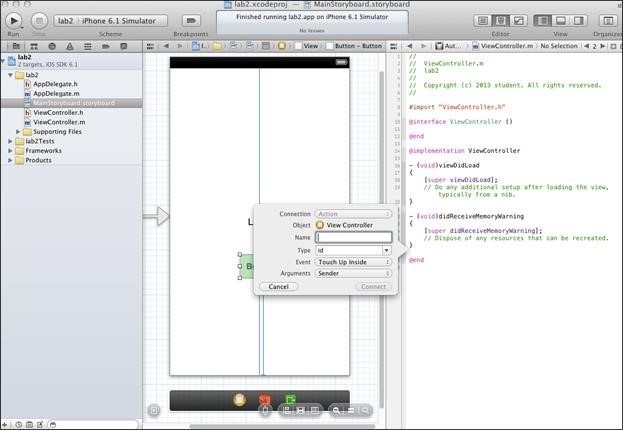
- (IBAction)refresh:(id)sender { }

**Листинг 2. Определение обработчика**

Добавьте в метод refresh код установки значения надписи (Листинг 2).

[[self indicator] setText:@"0 C"];





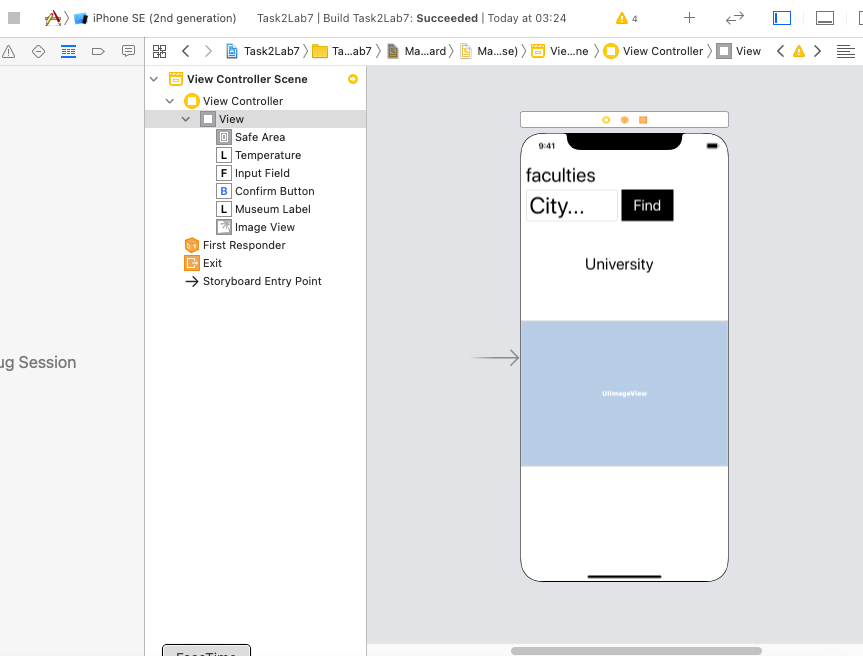
**Листинг 3. Установка значения надписи**

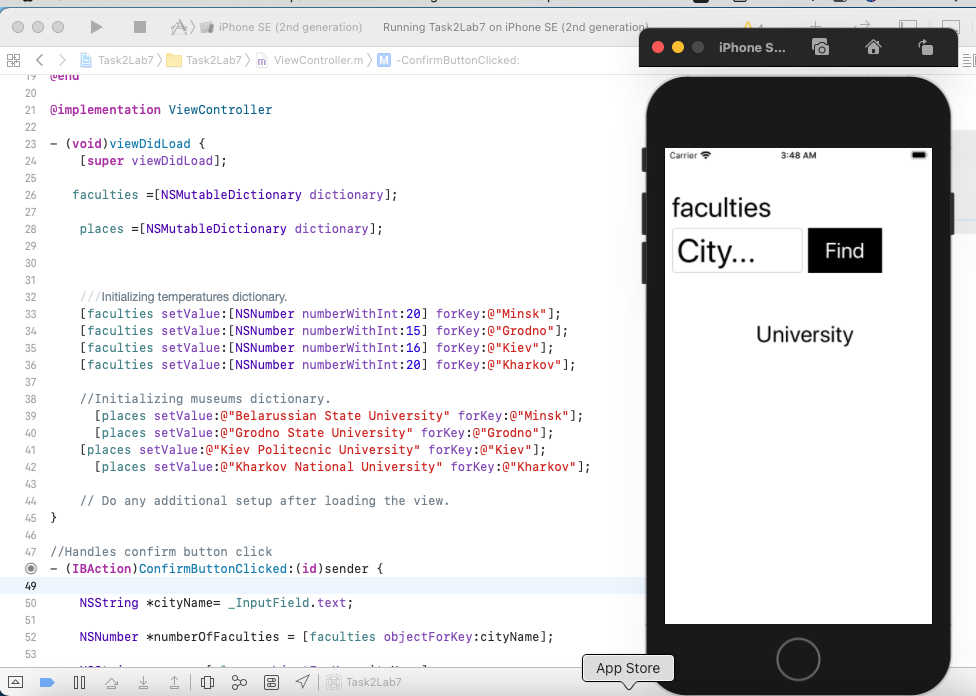
Запустите программу. Теперь при нажатии на кнопку надпись меняется на 0 C (Рис. 19).

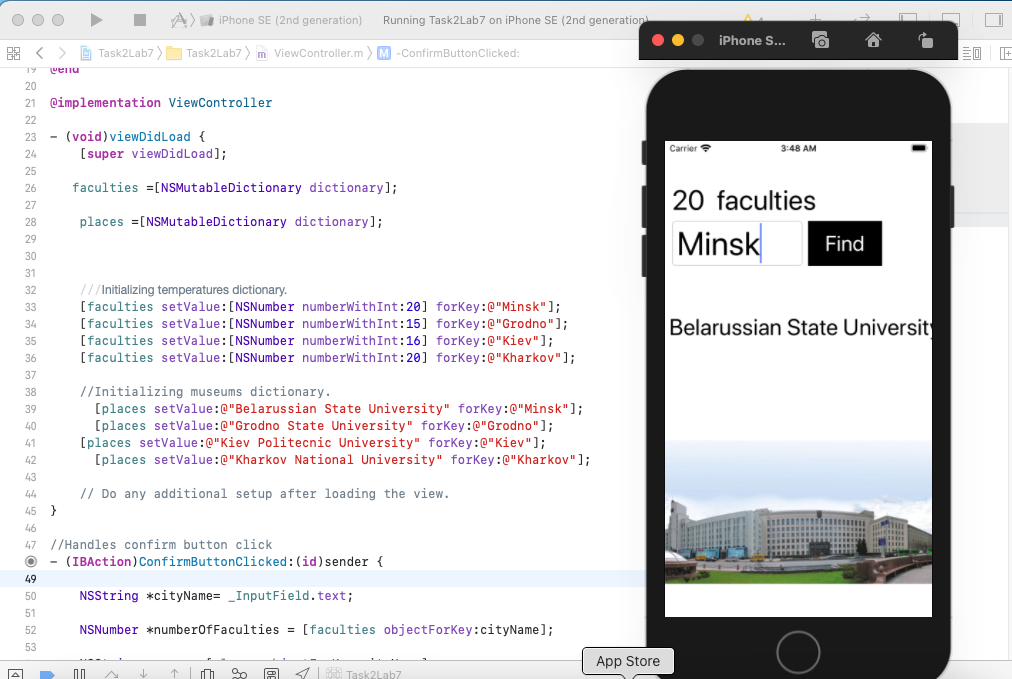
Теперь необходимо добавить функционал получения текущей температуры. (Листинг 3).

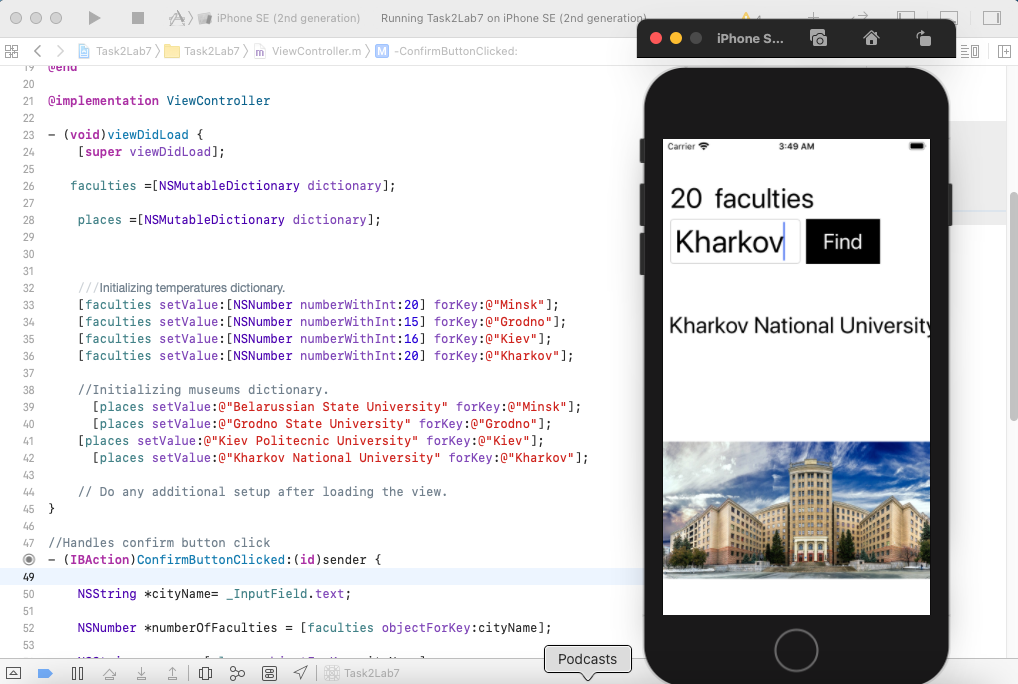


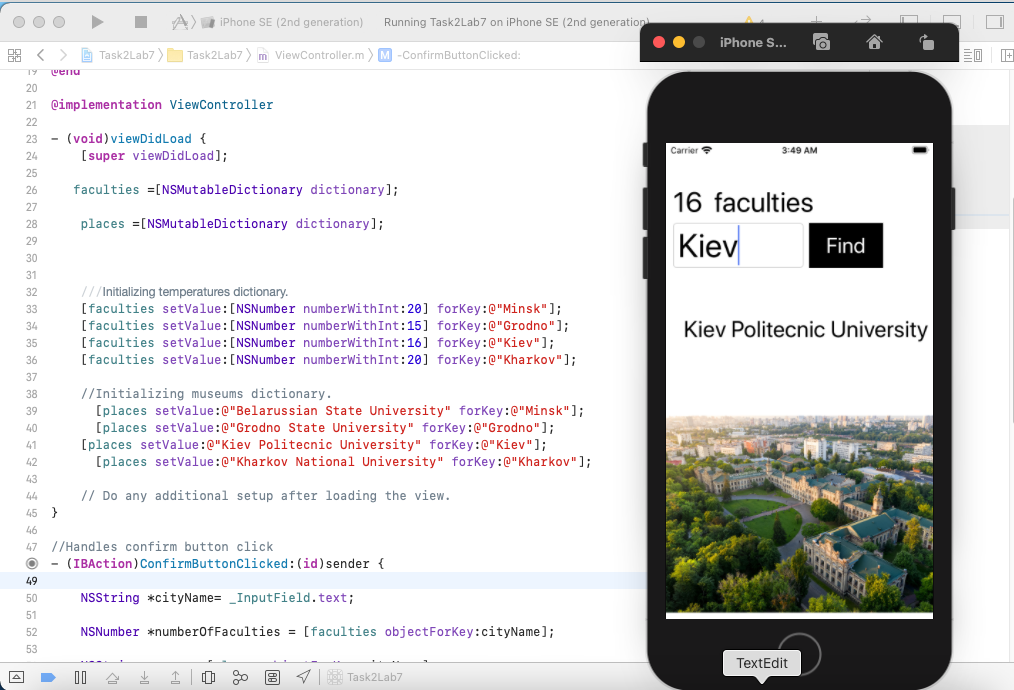
2.2. Дополните приложение на Objective-C из 2.1 следующим функционалом, при выборе города выводит изобаржение университета города, и количество факультетов и студентов в нем. В приложении должны быть не менее двух городов из каждой страны и не менее двух стран, одна из которых Беларусь и университет — БГУ.

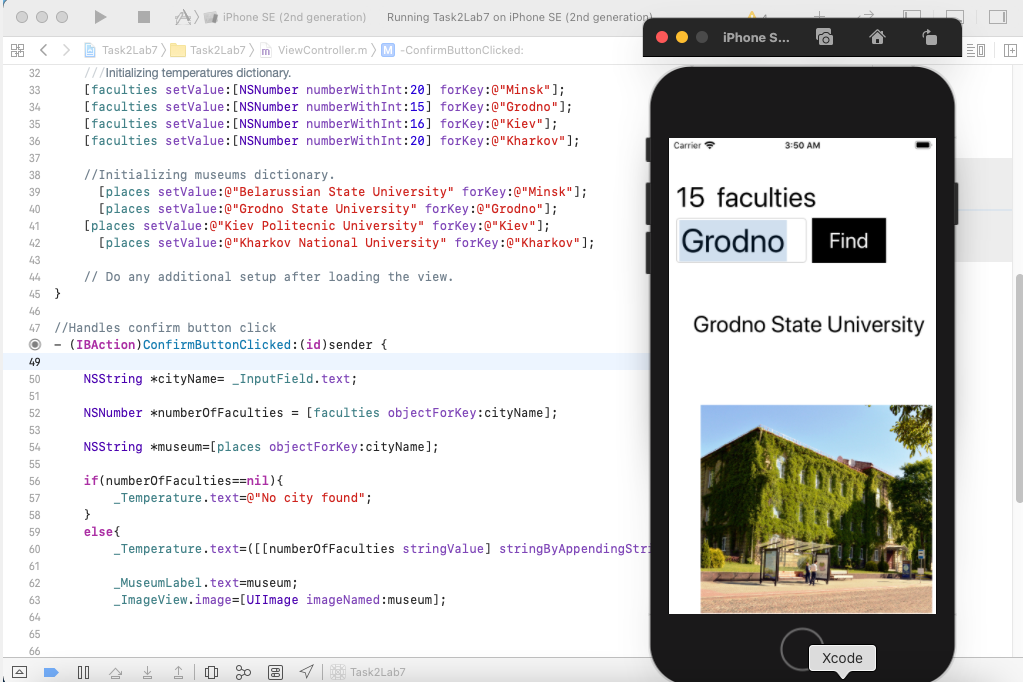










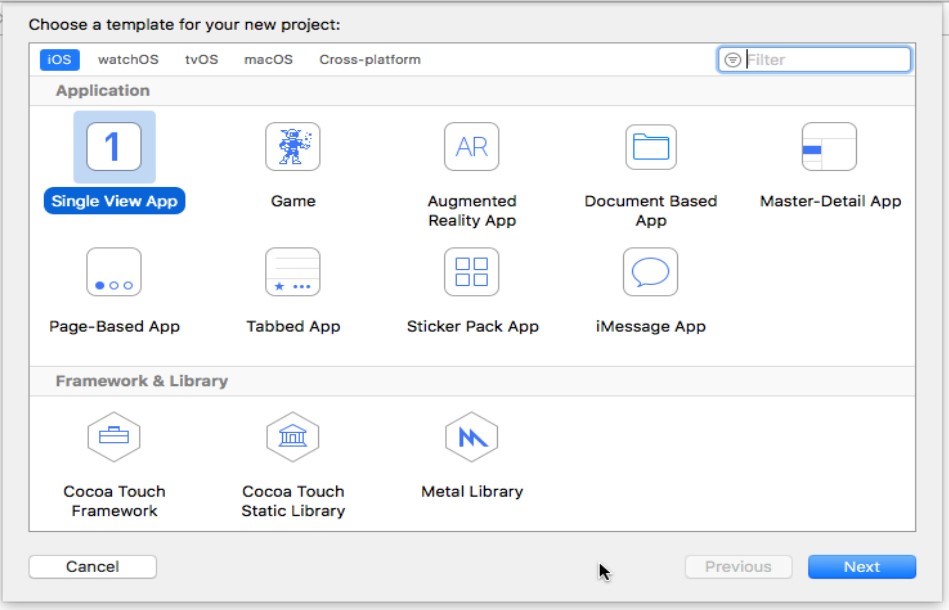


**ЗАДАНИЕ 3. ПРИЛОЖЕНИЕ ДЛЯ РАСЧЕТА КОЛИЧЕСТВА КАЛОРИЙ И ИНДЕКСА МАССЫ ТЕЛА НА ЯЗЫКЕ SWIFT**

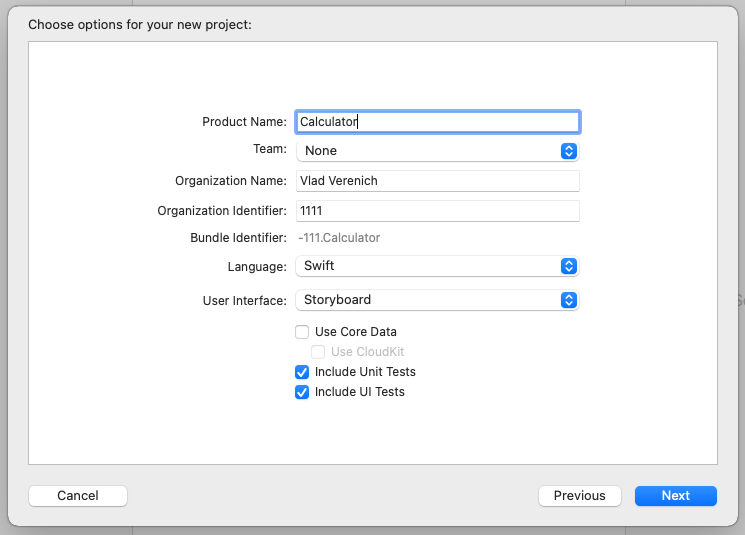
В данном задании реализовать простое приложение для расчета индекса массы тела и необходимого количества калорий для поддержания веса. Для расчет индекса массы тела использовать формулы Гарисса Бенедикта и индекса массы тела.

В руководстве приведён пример исходного кода для Swft 3. Адаптировать для Swift 5 и, при необходимости, исправить ошибки, связанные с версией языка.

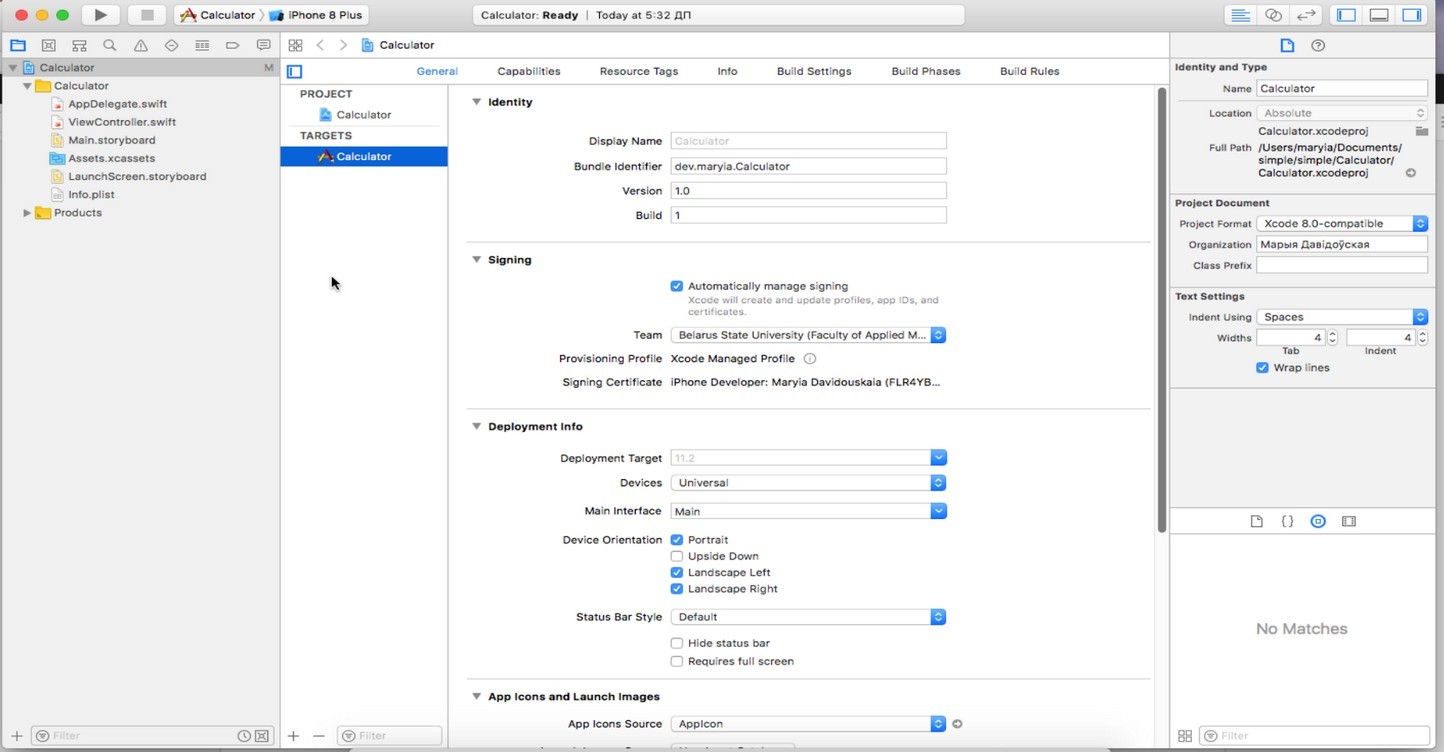
1. Для начала давайте создадим проект: запустите Xcode и нажмите комбинацию клавиш CMD+Shift+N или выберите в меню File→New→Project



1. Теперь выберите Single View Application (App) и нажмите Next.
2. На этом этапе надо придумать имя для приложения (Product Name), нажать Next и выбрать папку для сохранения. Главное, чтобы значение в поле Language было равно Swift.

****

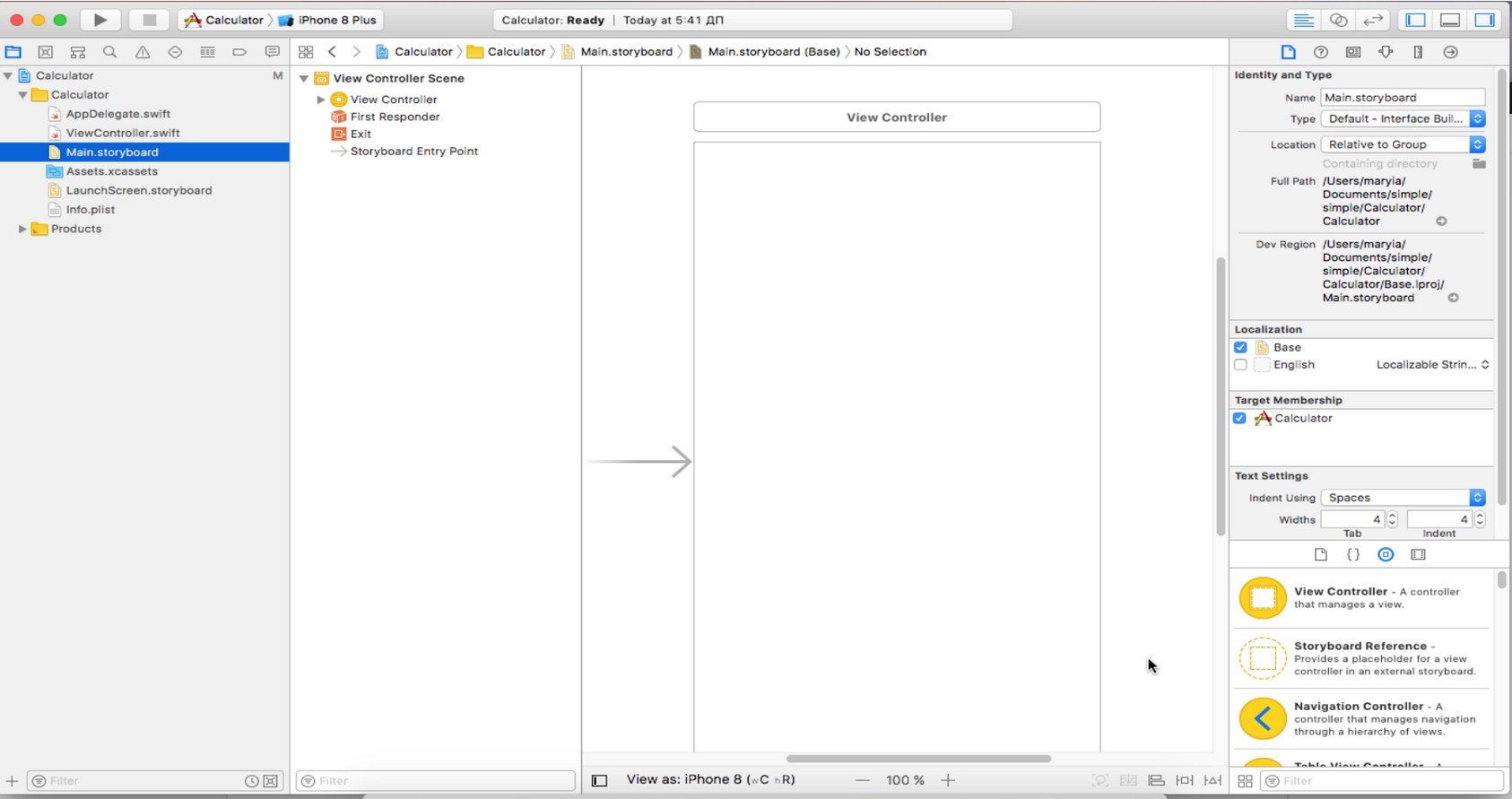
1. После того, как вы сохраните проект, перед вами появится рабочее пространство Xcode.



Слева у вас будет панель Navigator, а справа — Utilities. В каждой из этих панелей есть свои вкладки, которые позволяют получить доступ к различным функциям.

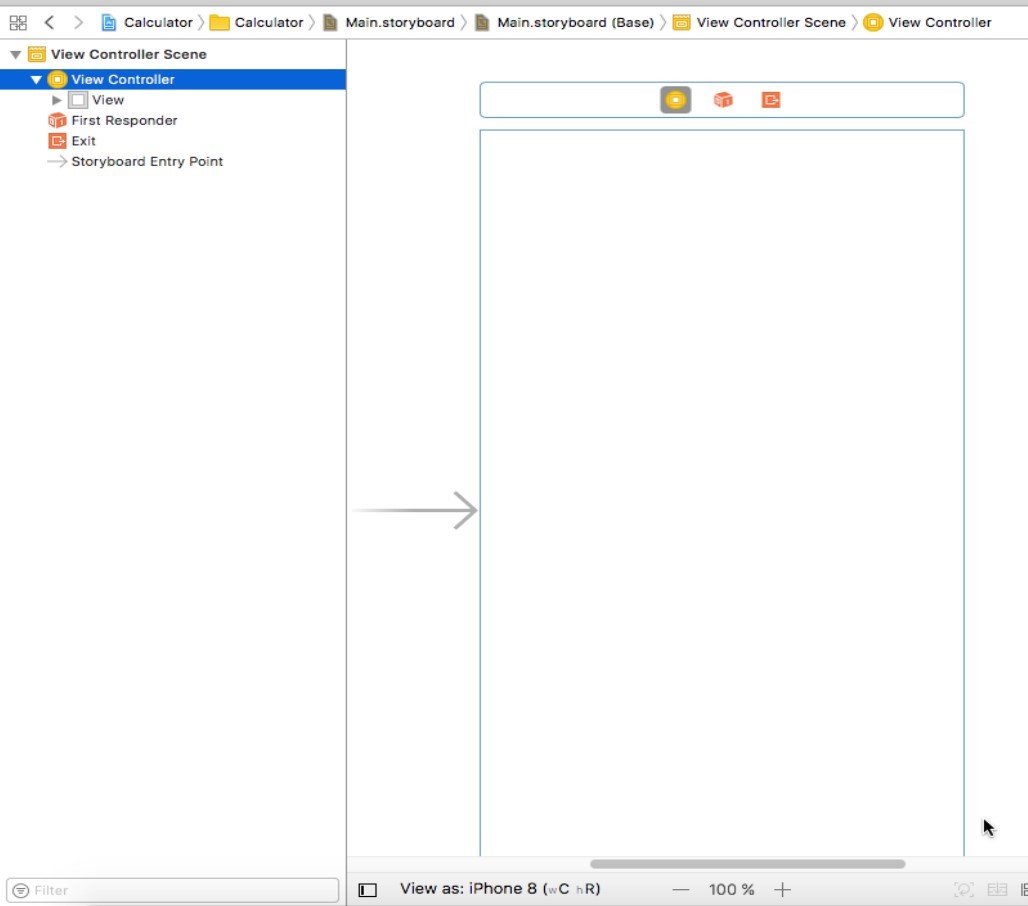
Например, первая вкладка в панели Navigator показывает список файлов проекта, третья позволяет производить поиск по проекту и так далее.

1. Найдите в списке файлов проектов файл с именем Main.storyboard и нажмите на него.

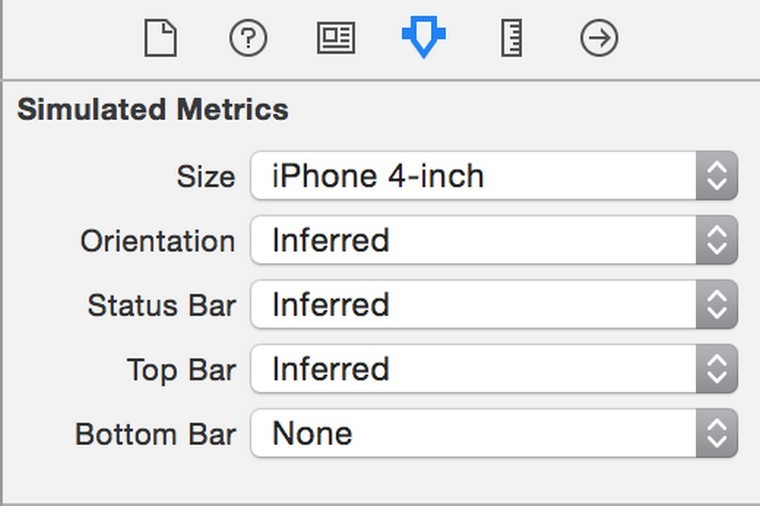


Файл Main.storyboad определяет, какие экраны (контроллеры) есть в приложении. Вы можете добавлять элементы на экраны, задавать связи между экранами и так далее.

Если вы выберите какой-нибудь контроллер, он подсветится синей рамкой:



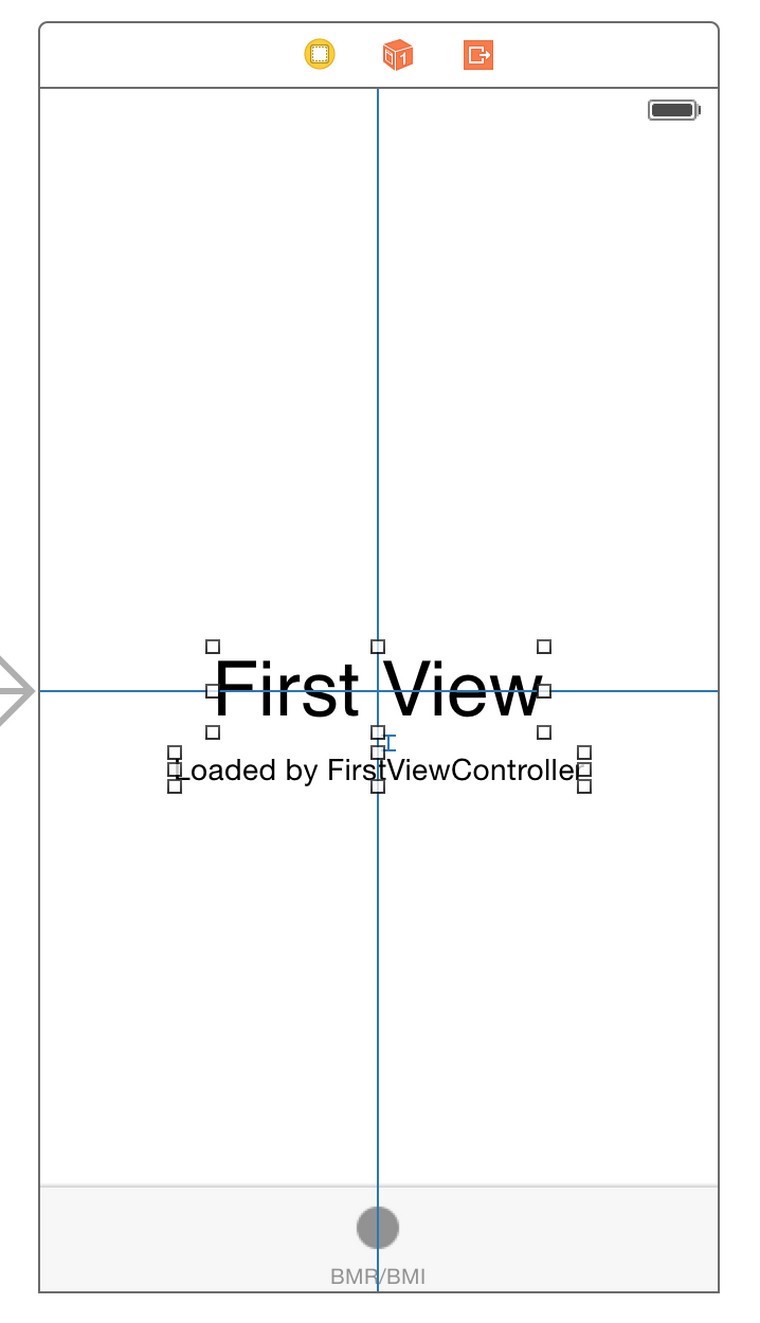
После того, как контроллер выбран, его свойства начнут отображаться в панели Utilities. Например, контроллеру можно поменять размер, выбрав другое значение в поле Size.



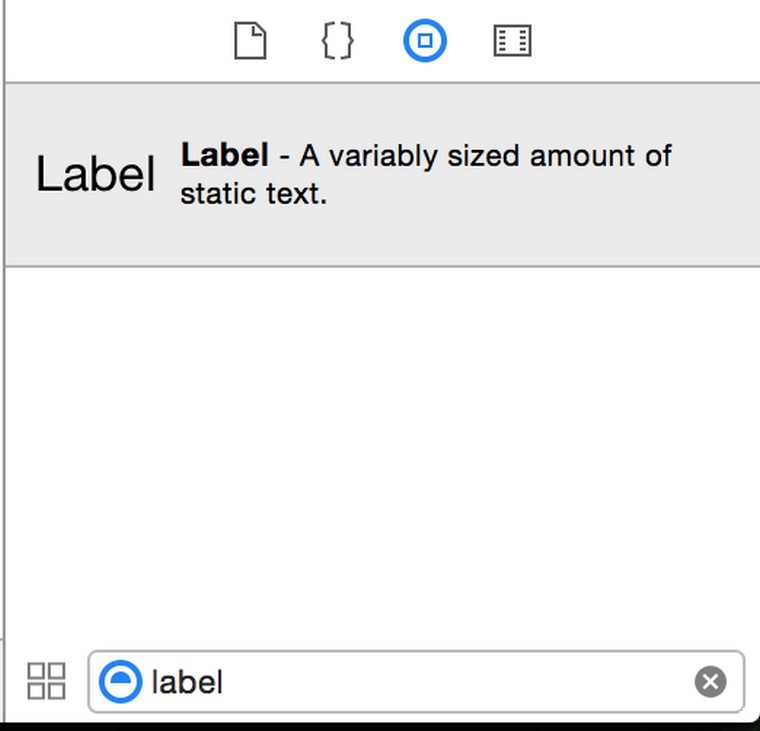
1. Запустим наше шаблонное приложение и посмотрим, как оно выглядит в симуляторе. Для этого надо в верхнем левом углу выбрать тип симулируемого устройства и нажать комбинацию клавиш CMD+R или кнопку с иконкой Play.



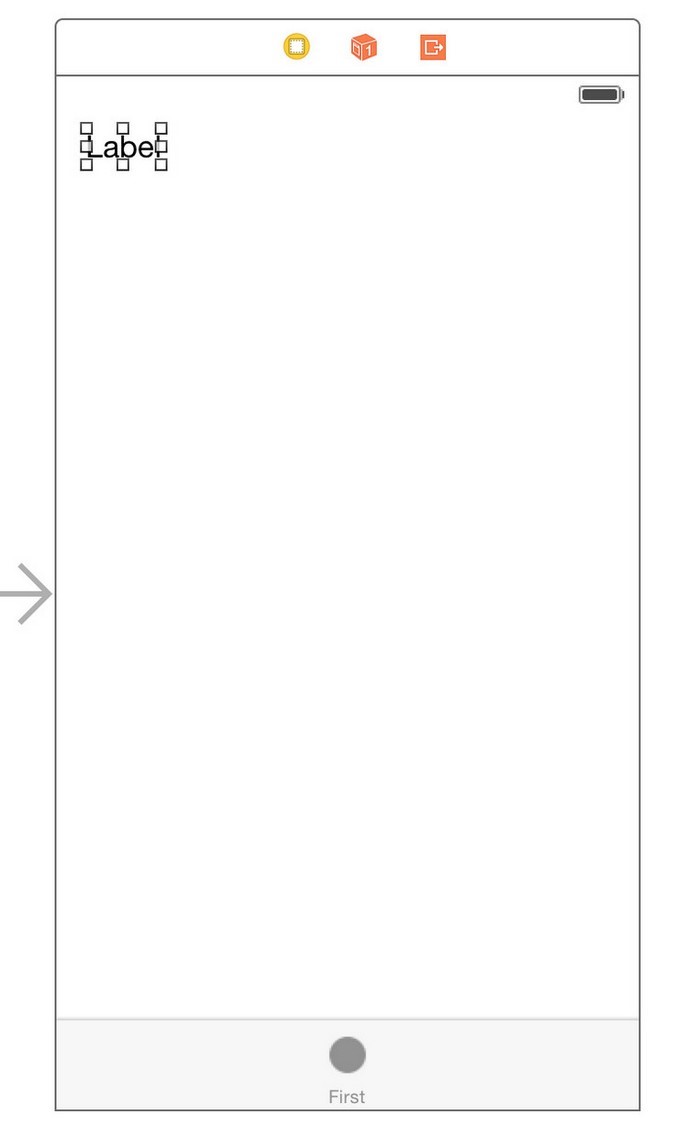
Как видите, у нас есть приложение с двумя экранами, переключение между которым осуществляется при помощи панели вкладок.



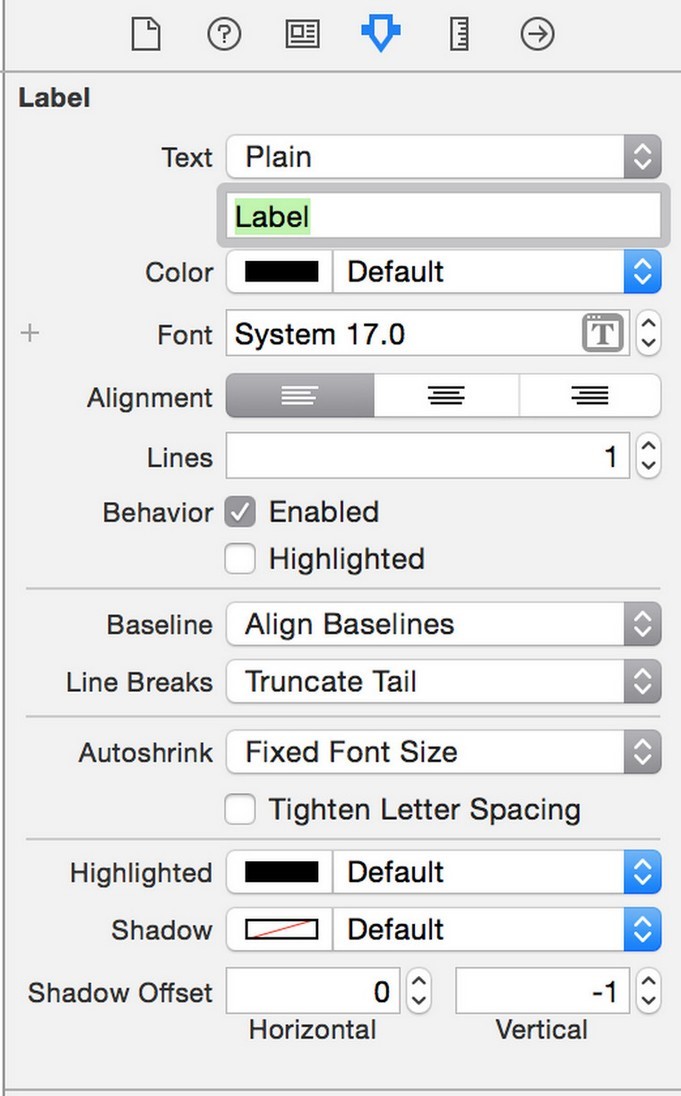
1. Теперь добавим свои элементы управления. Внизу панели Utilities есть Object Library, в которой можно выбрать объекты и добавить их к сцене. Найдите объект Label и перетащите его на контроллер BMR/BMI. Перед этим не забудьте два раза тапнуть на сцене, чтобы масштаб стал стандартным.



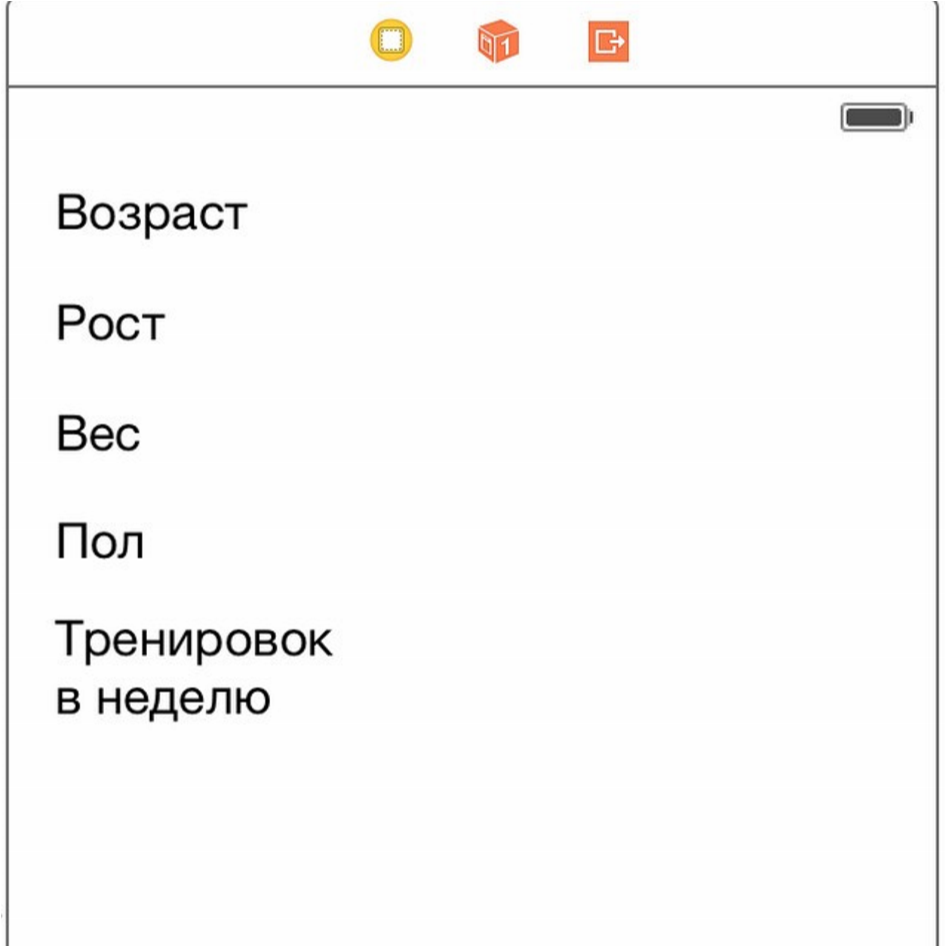
Должно получиться примерно так.



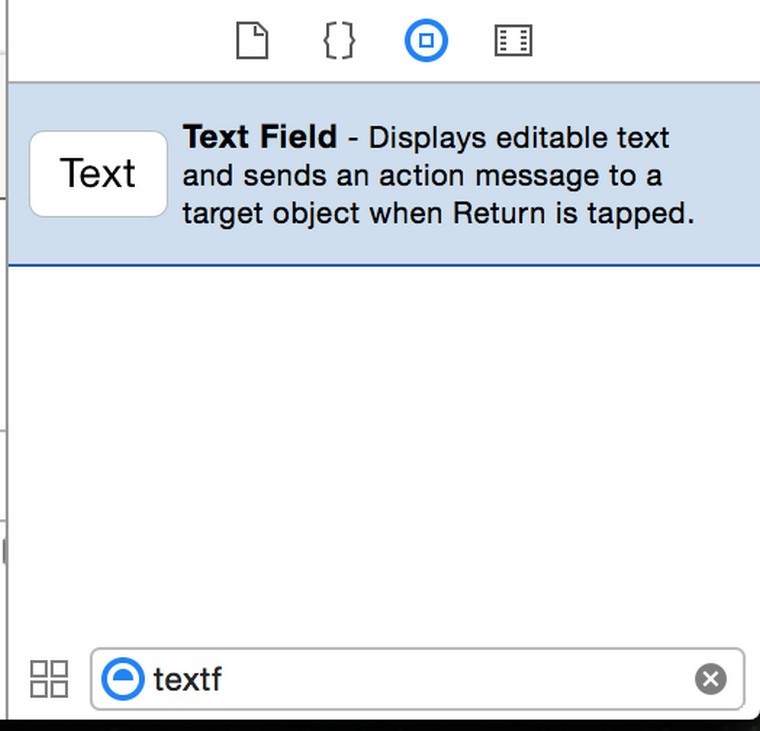
В панели Utilities можно поменять текст, размер шрифта и так далее.



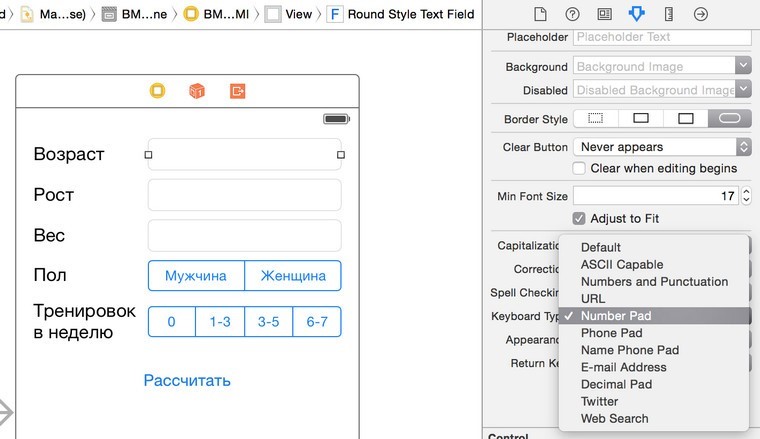
1. Перетащите еще несколько лейблов и поставьте им такой же текст, как и на картинке. Для последнего лейбла нужно поставить Lines в 2, чтобы текст переносился на другую строку.



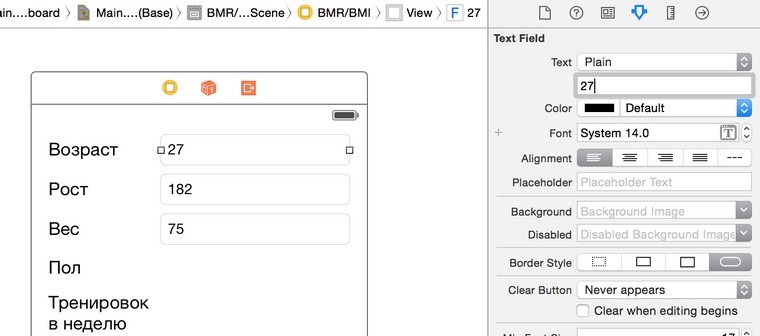
Теперь добавьте 3 текстовых поля (Text Field).



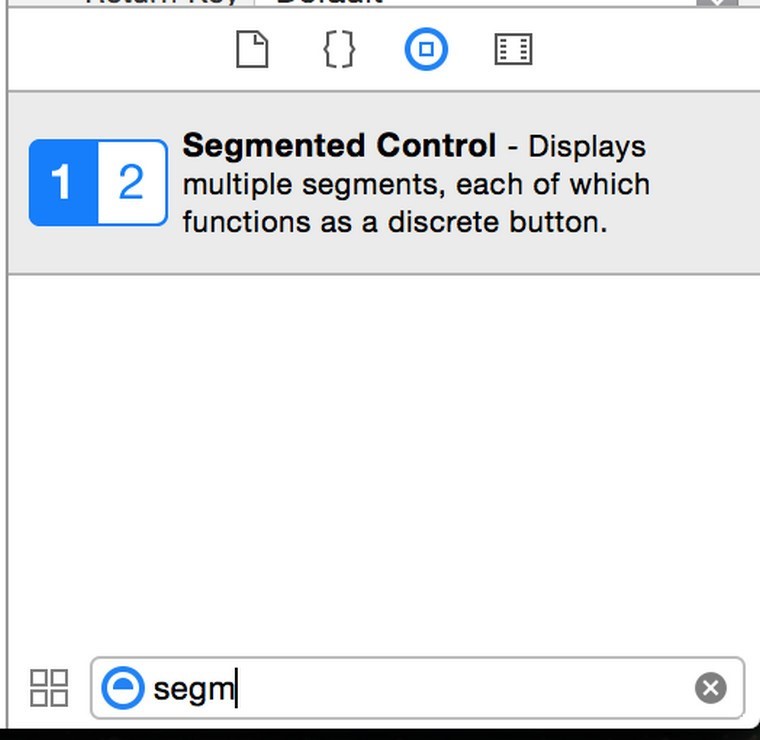
И разместите их, как на картинке. У текстовых полей поставьте параметр keyboard type в Number Pad.



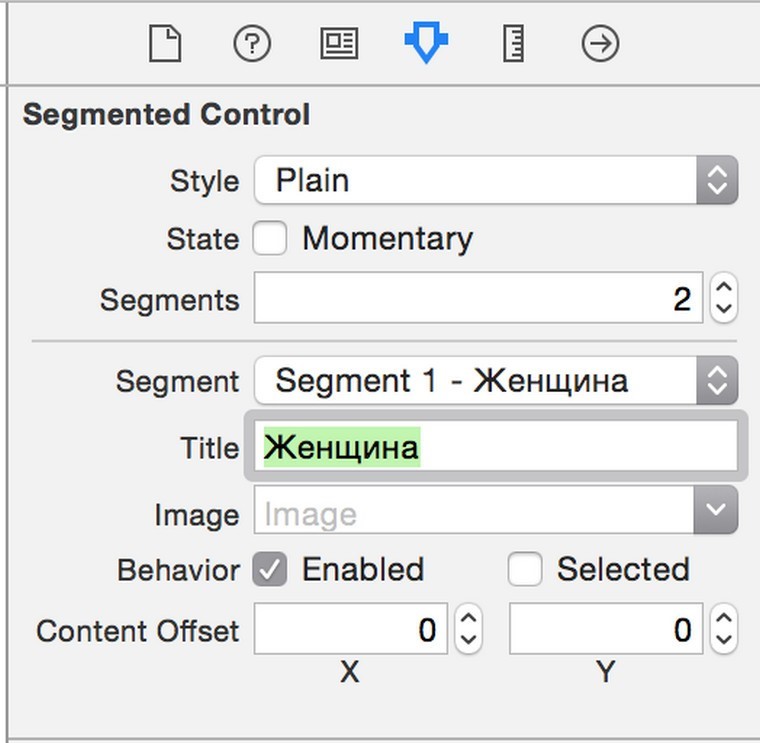
И установите текст по умолчанию.



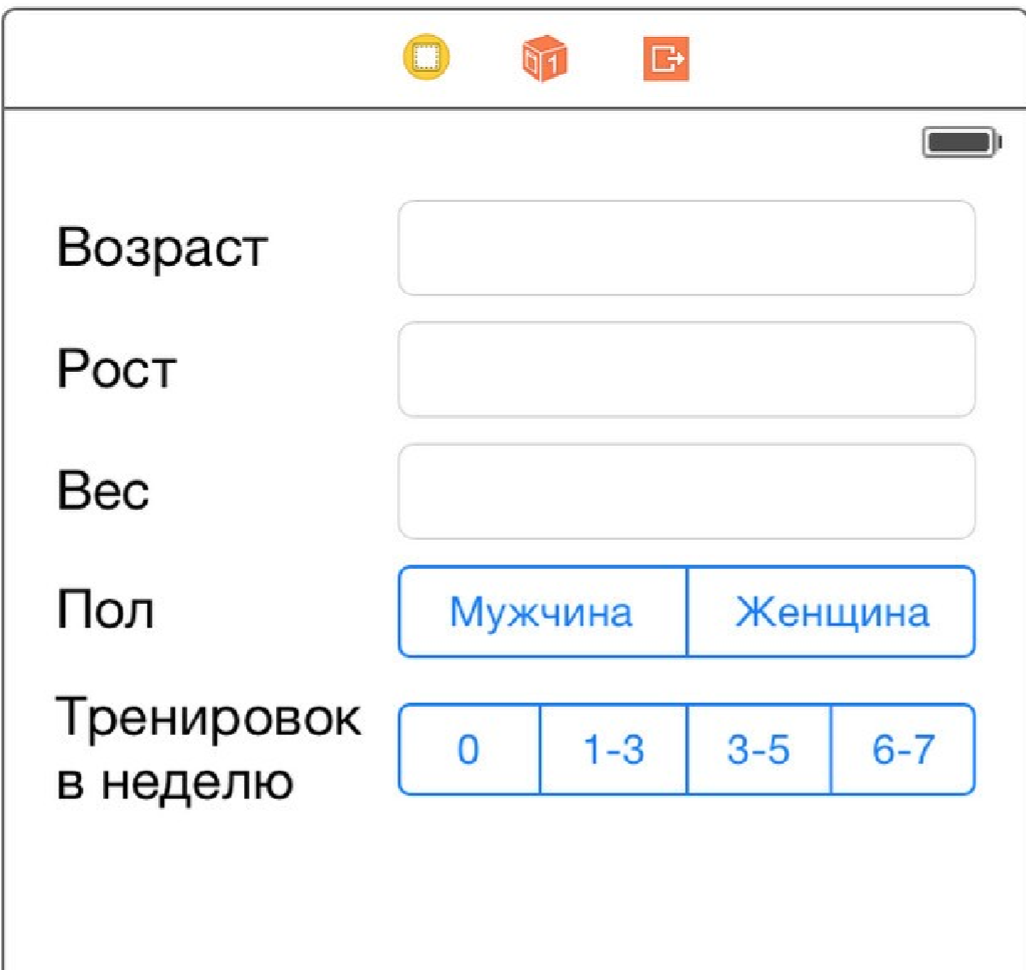
1. Теперь добавим элементы управления для выбора пола и кол-ва тренировок в неделю. В Object Library ищете Segmented Control и добавляйте на экран.



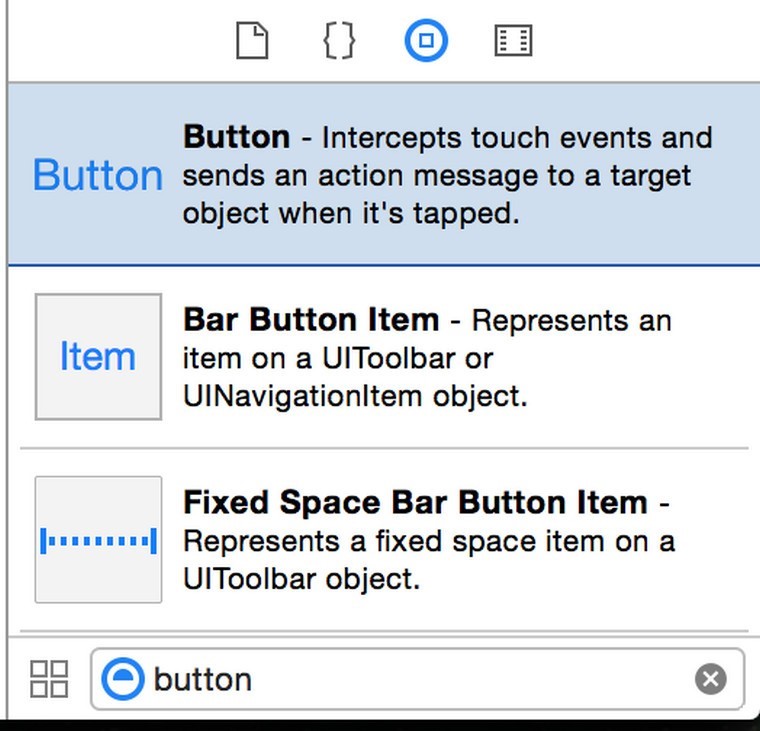
У Segmented Control можно менять количество сегментов и текст для каждого сегмента.



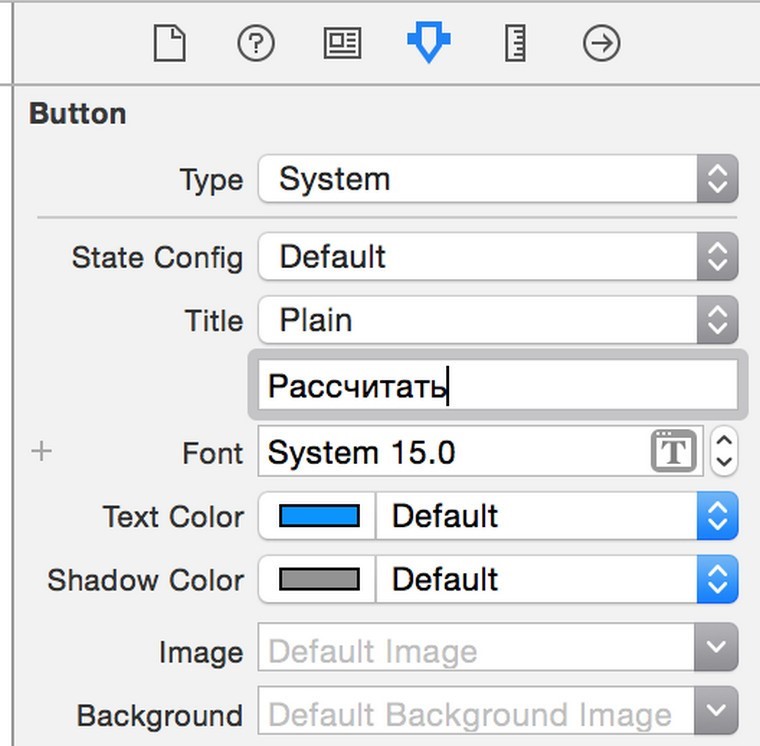
Сделайте так, чтобы сегменты выглядели, как на скриншоте.



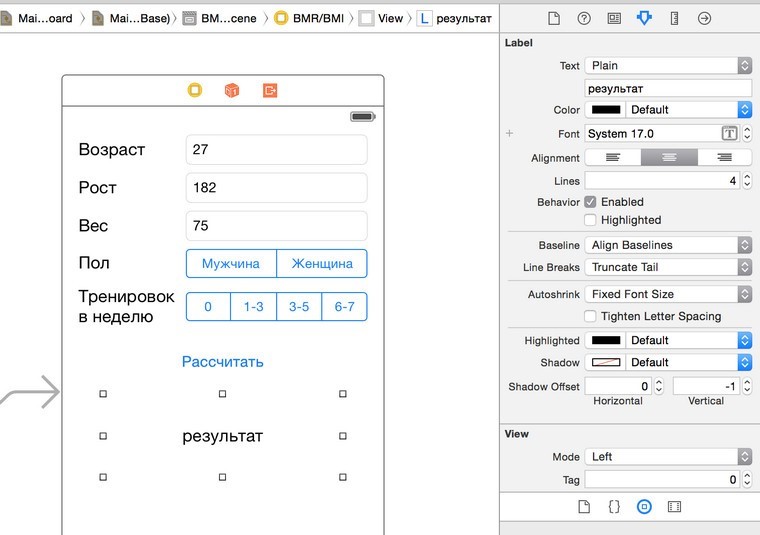
1. Теперь добавьте кнопку (button).



И установите ей заголовок.



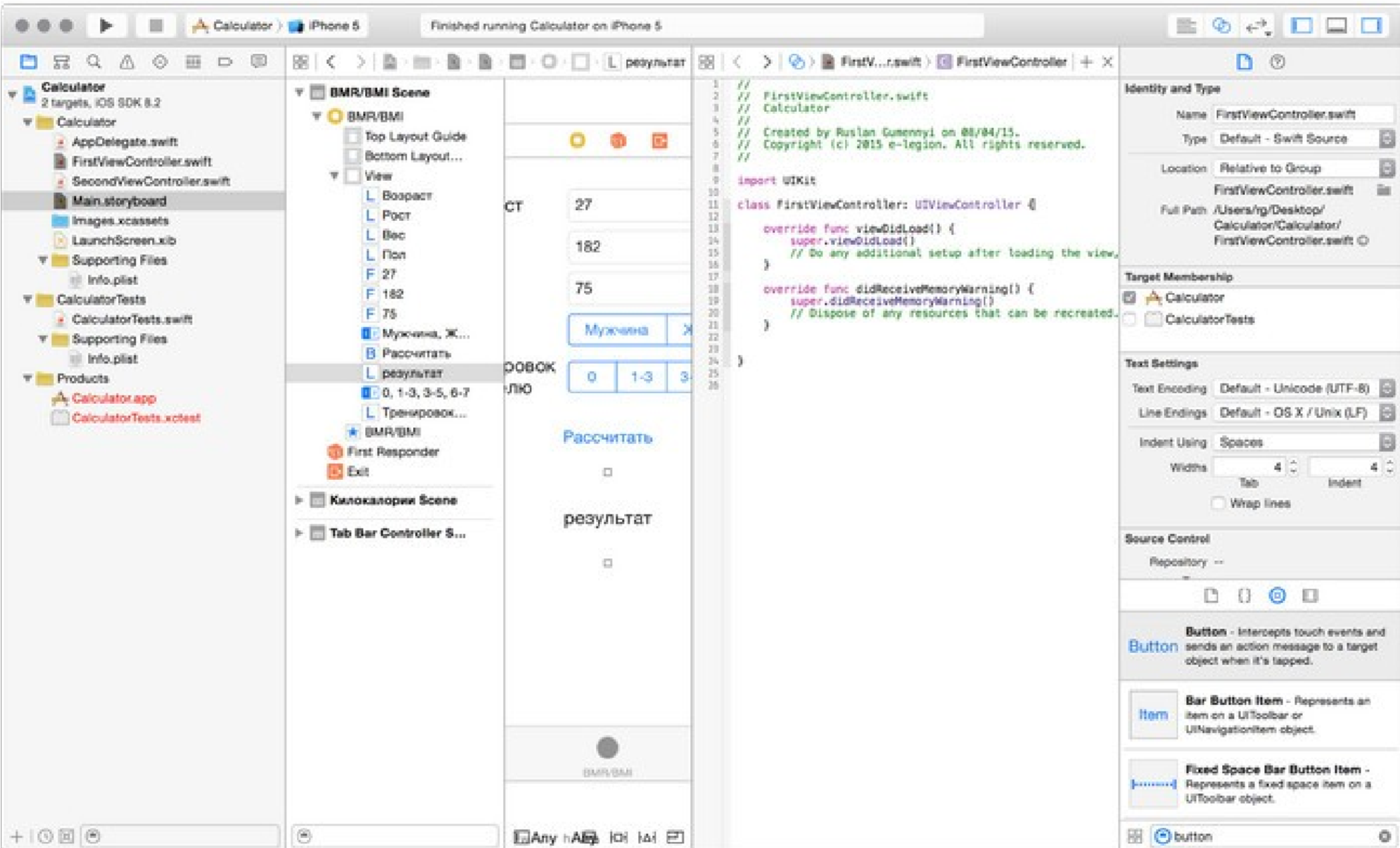
После этого добавьте еще один label с lines равным 4.



1. Выберите контроллер BMI/BMR, нажмите два пересекающихся кольца в правом верхнем углу и откроется Assistant Editor. Он показывает код, ассоцированный с этим контроллером

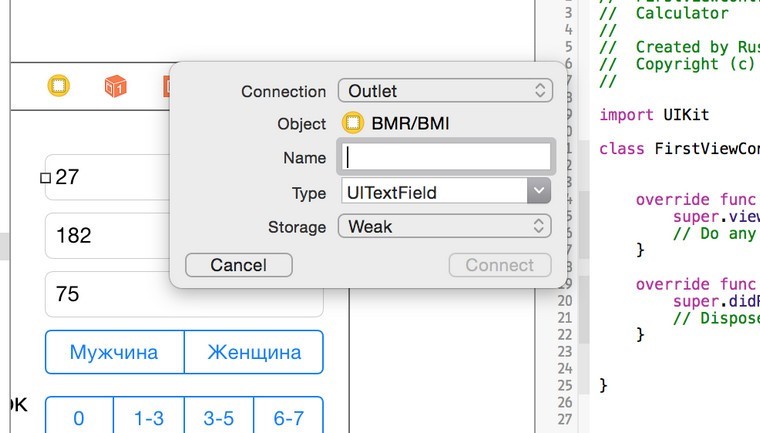


1. Протащим элементы управления.

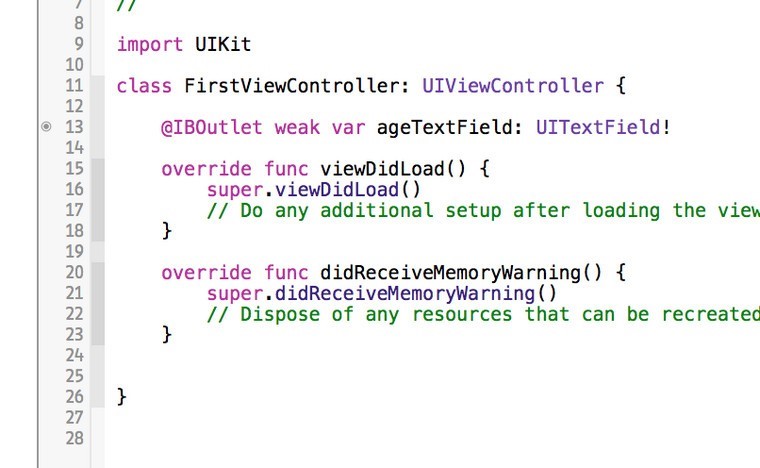


Для этого выберите первый text field (в нашем случае это будет возраст), нажмите Ctrl, кликните на него еще раз и, не отпуская кнопки, перетащите внутрь класса. Если все сделано правильно, то вы увидите подсказку “Insert Outlet, Action, …”.

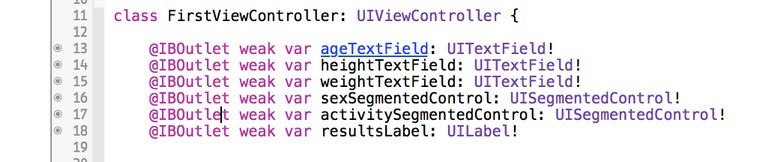
Теперь отпустите курсор, и вы увидите окошко для создания связи. В качестве имени введите ageTextField и нажмите Connect.



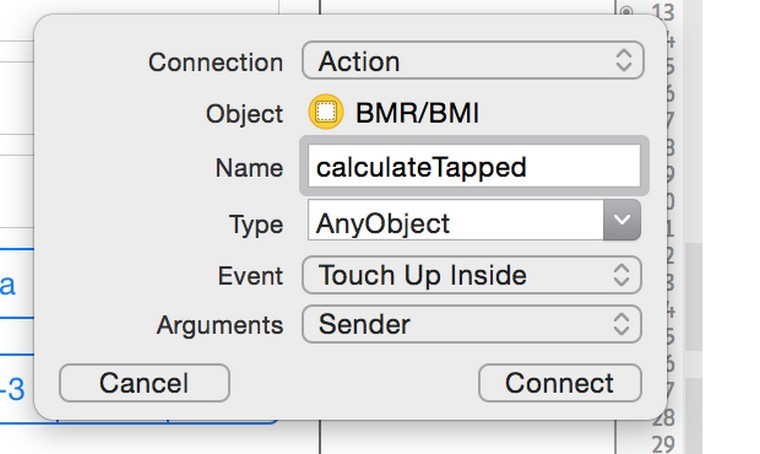
1. У вас появится переменная ageTextField.



Проделайте эту процедуру для оставшихся text field, segmented controls и label с текстовым результатом. У вас должно получиться так:

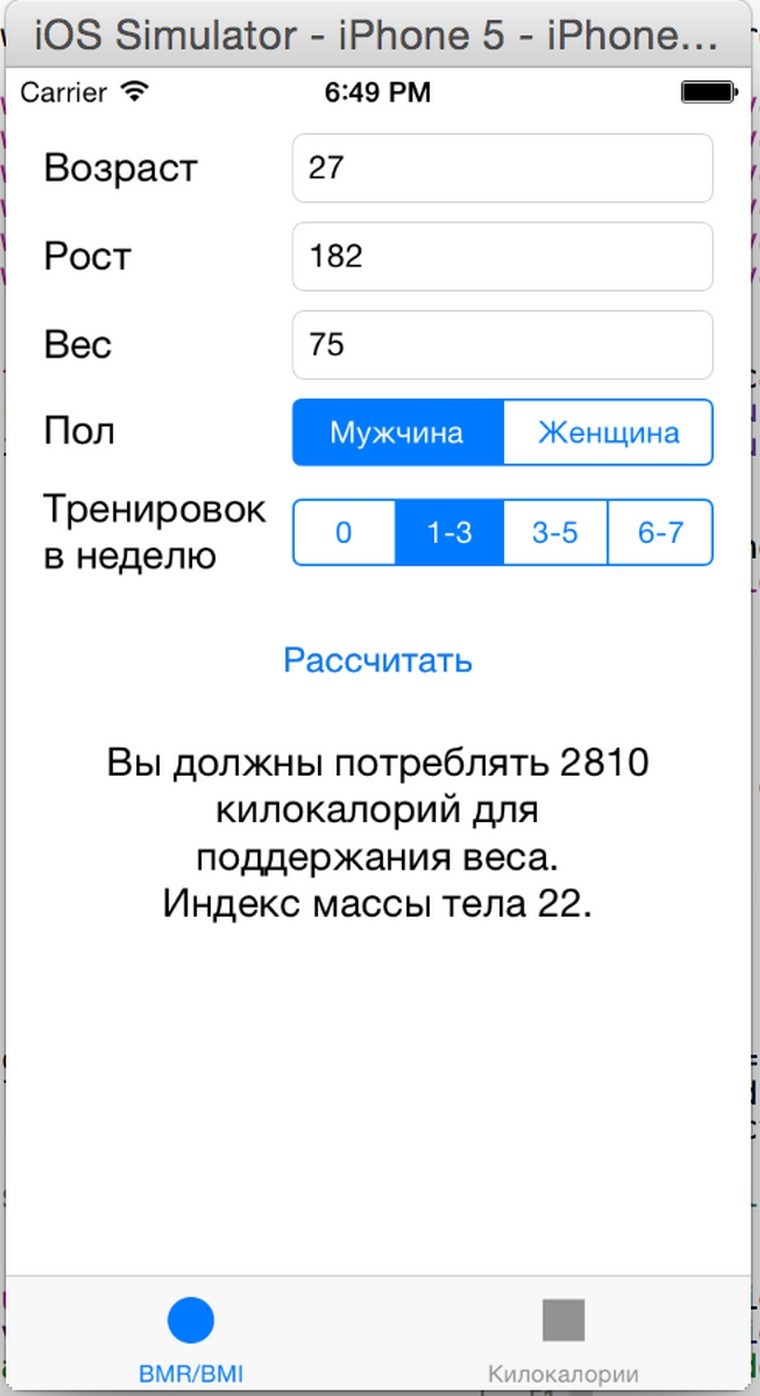


Теперь протащите кнопку, но тип соединения укажите не Outlet, а Action. И в качестве имени используйте calculateTapped.

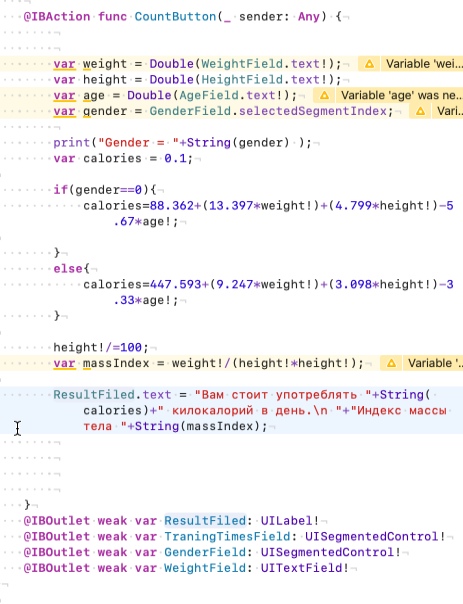




1. Добавьте реализацию расчёта показателя BMR и индекса массы тела в метод calculateTapped.
2. Проверьте результат.



Код:



**ЗАДАНИЕ 4. МОБИЛЬНОЕ ПРИЛОЖЕНИЕ С ИСПОЛЬЗОВАНИЕМ SWIFTUI НА ЯЗЫКЕ SWIFT** Разобрать и реализовать пример из статьи «SwiftUI: Getting Started»

ЗАДАНИЕ 5. ПРОСТОЕ МОБИЛЬНОЕ ПРИЛОЖЕНИЕ НА ЯЗЫКЕ SWIFT

Используя изученные типы полей в заданий 3, реализовать простое приложение на языке swift с использванием UIKit или SwiftUI (на выбор) согласно варианту из 2-х сцен. Адаптировать приложение для книжной и альбомной ориентации и под разные размеры экранов.

Продемонстрировать использование типов полей text field, segmented controls и label

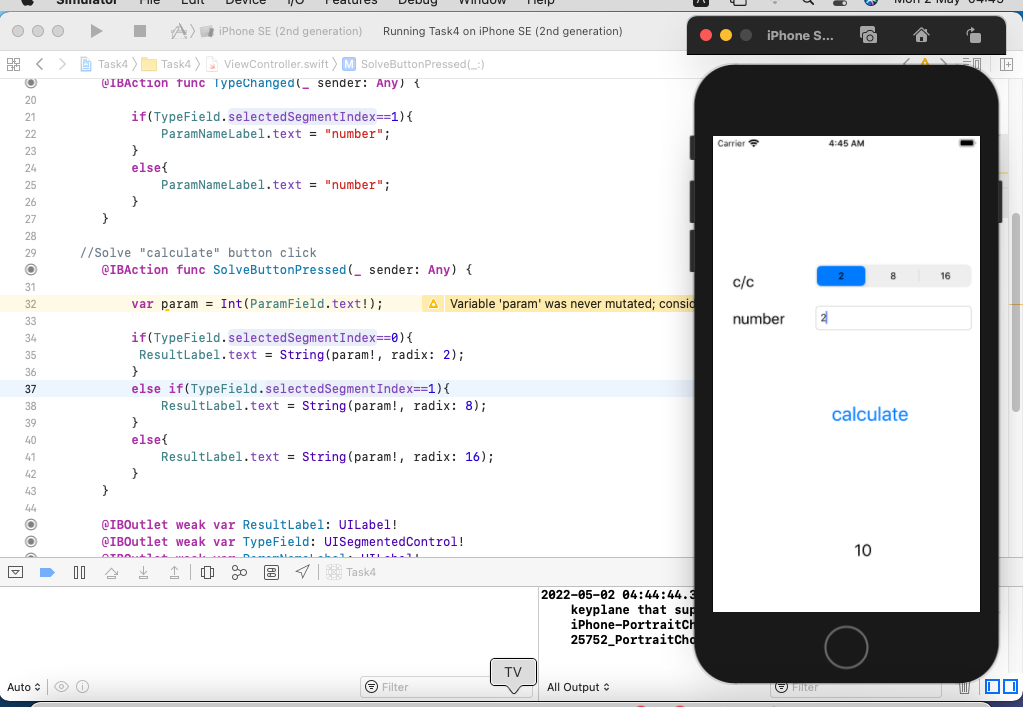
При создании приложения с помощью UIKit и InterfaceBuilder продемонстрировать позиционирование элементов с помощью AutoLayout и StackView, которые обеспечивают адаптацию приложения под разные экраны и ориентацию. Пример приложения с использованием AutoLayout, StackView и нескольких сцен представлен в статье https://www.twilio.com/blog/2018/05/xcodeauto-layout-swift-ios.html. Рекомендуется для изучения!!!

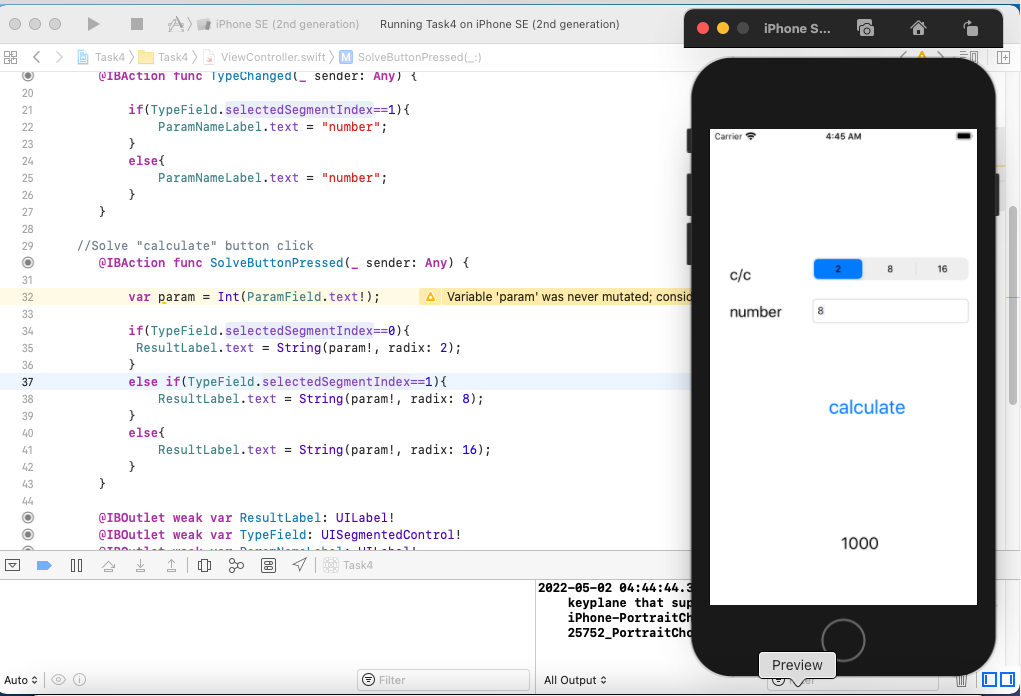
При написании кода следовать рекомендациям стиля для языка Swift и добавить комментарии :

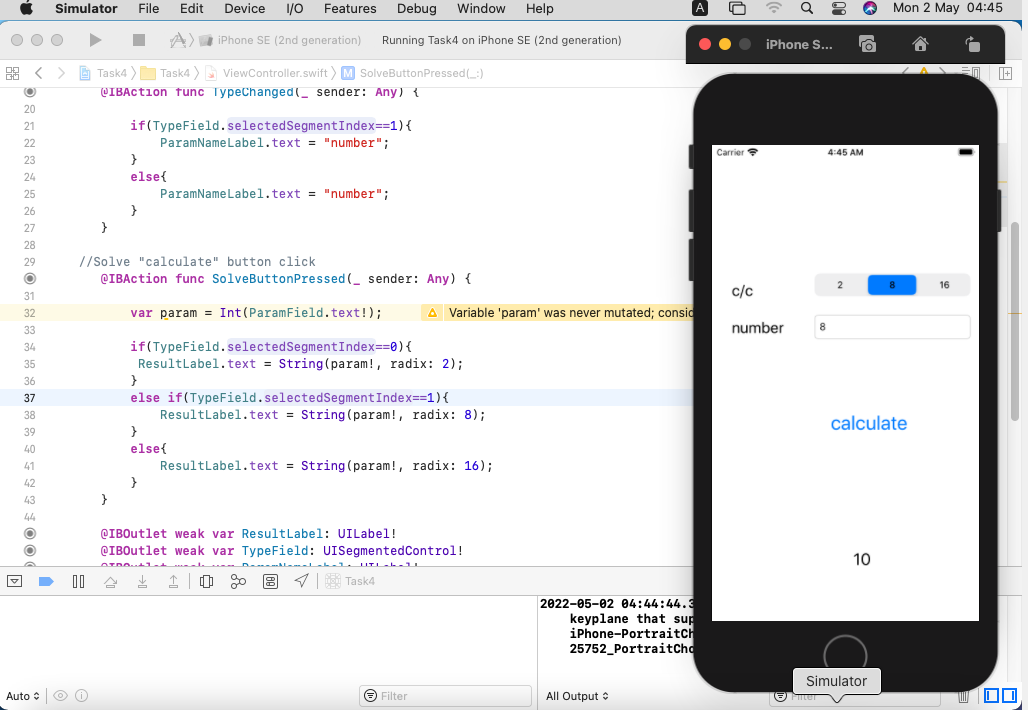
a) <http://ilya2606.ru/?p=1846>

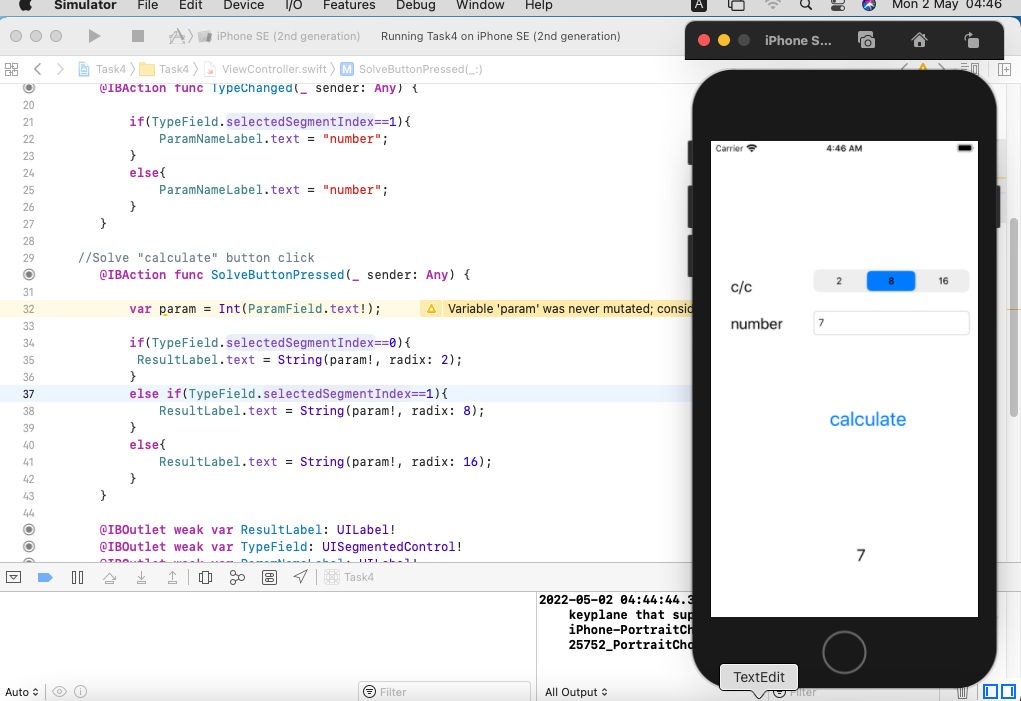
b) <https://github.com/RedMadRobot/RMR-swift-style-guide>

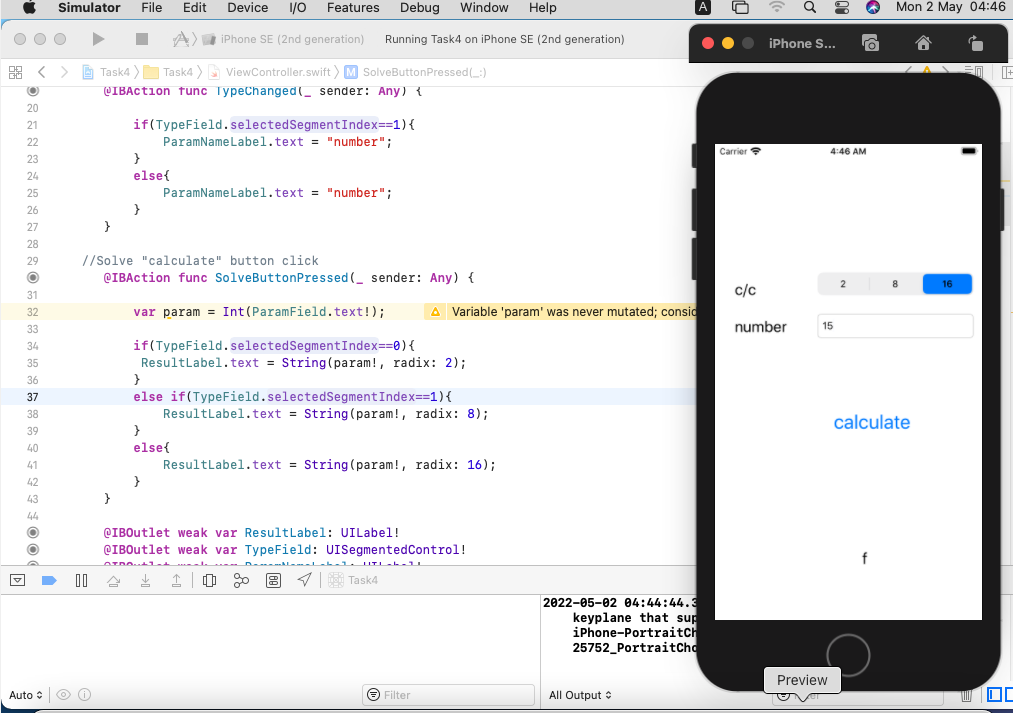
c) <https://github.com/raywenderlich/swift-style-guide>

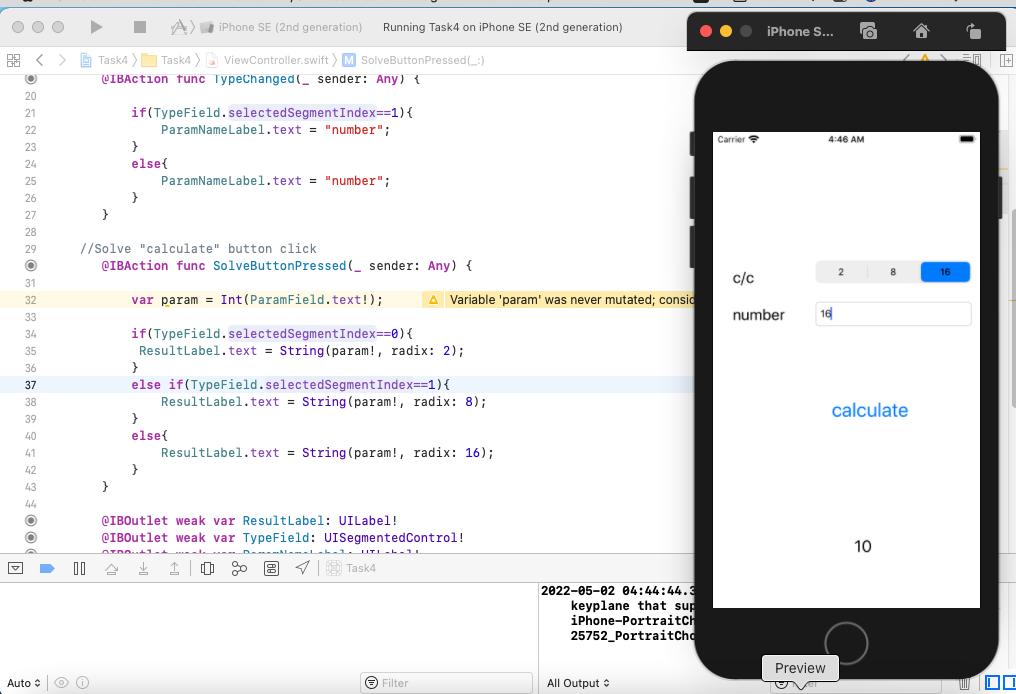


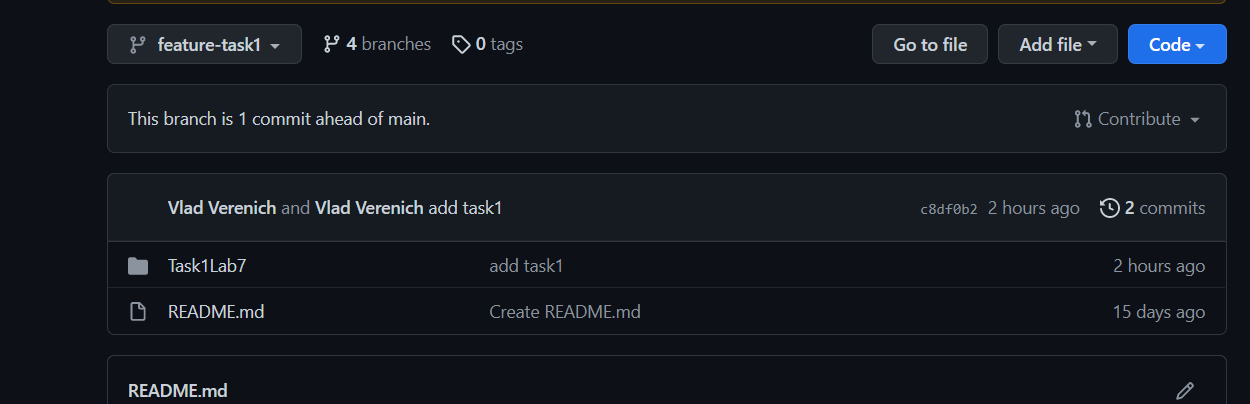


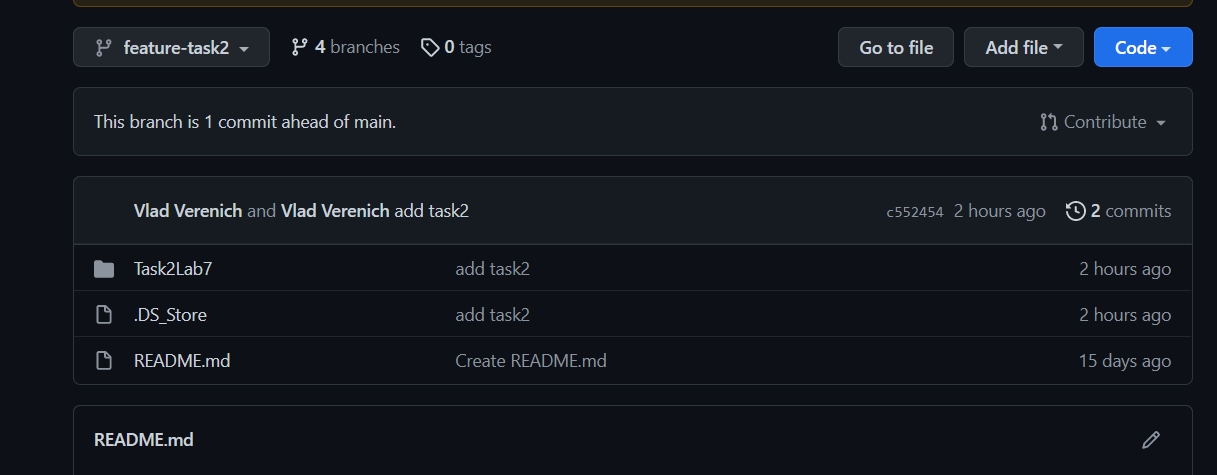


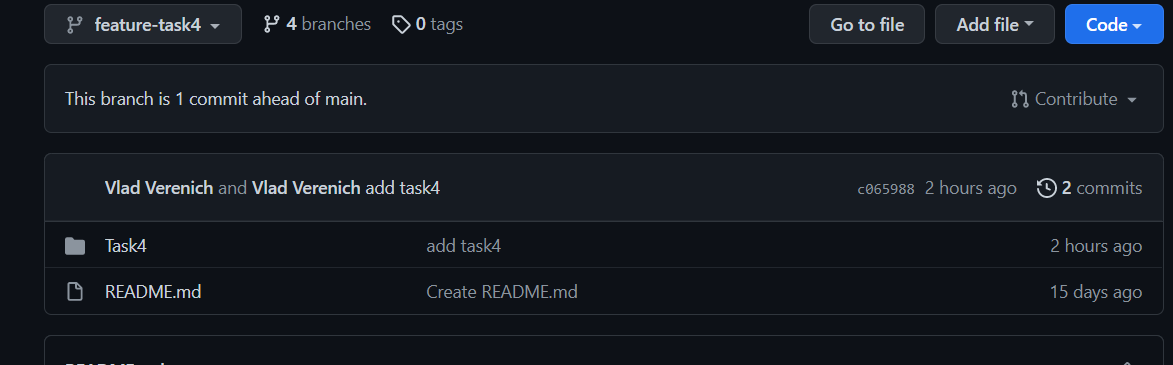


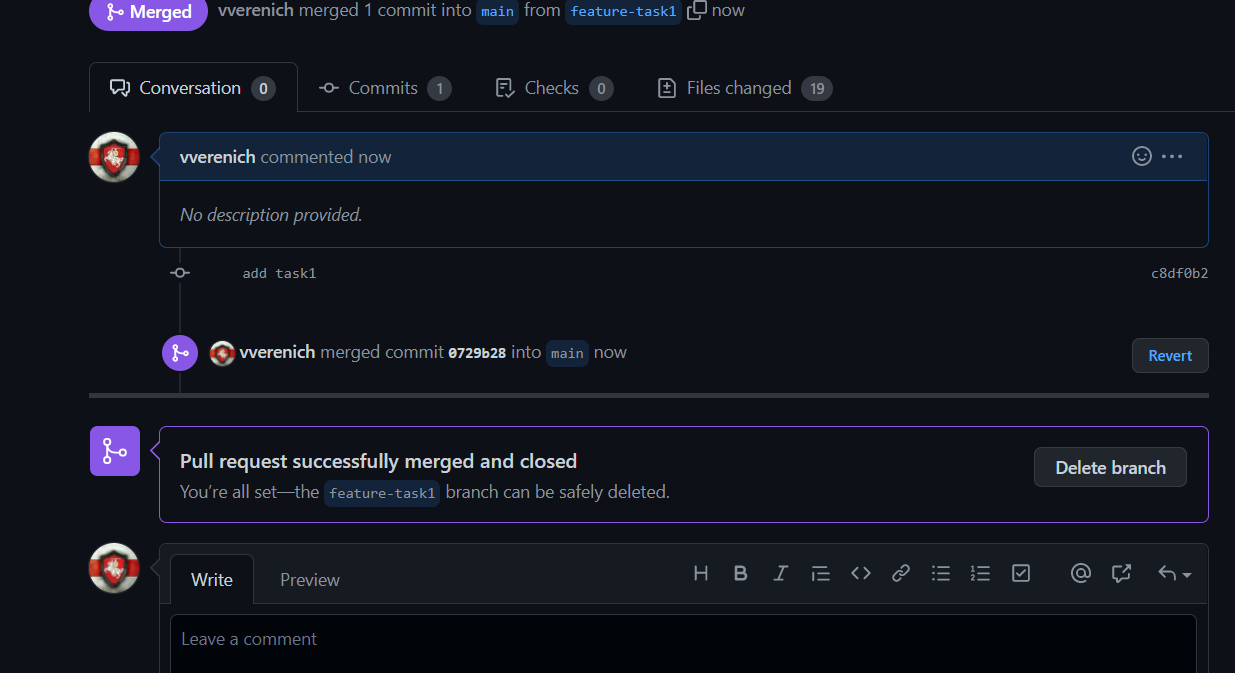


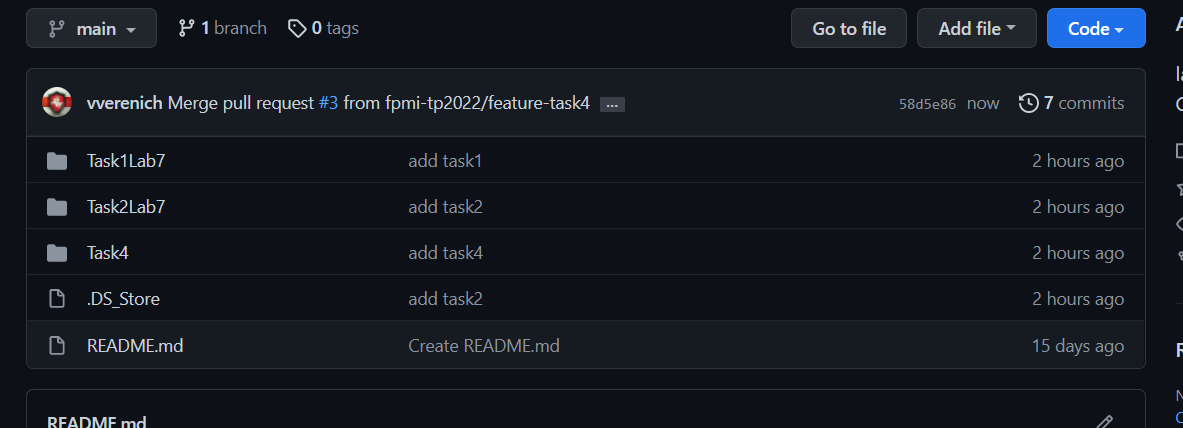












**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Как зарегистрироваться как разработчик apple и добавить идентификатор организации, под которым будут создаваться приложения в профиль разработчика и xcode?

**ОТВЕТ:**

* 1. Запускаем Xcode. В первый раз у нас появляется возможность добавить аккаунт разработчика.
  2. Принимаем предложение и используя свой appleID регистрируем бесплатный или платный аккаунт разработчика.

1. Как добавить элементы управления на экран (контроллер)?

**ОТВЕТ:**

Сочетание клавиш shift+cmd+L выведет окно всех доступных элементов. Перетаскиваем любой понравившийся элемент на необходимый ViewContoller. Готово.

1. Какие типы элементов управления есть в библиотеке объектов Xcode?

**ОТВЕТ:**

UIButton

UIDatePicker

UIPageControl

UISegmentedControl

UISlider

UIStepper

UISwitch

1. Что такое IBOutlet?

**ОТВЕТ:**

Макрос, обозначающий некоторый объект, соответствующий определенному UI элементу.

1. Что такое IBAction?

**ОТВЕТ:**

Макрос, обозначающий некоторую функцию, соответствующий определенному событию вызванному определенным UI элементом.

1. Опишите способы создания IBOutlet и IBAction?

**ОТВЕТ:**

1. Написать вручную, а после этого путем перетягивания связать с необходимым событием или UI – объектом.
2. Нажать правой кнопкой мыши оп UI элементу. После чего появится выбор , того что можно для него создать, в том числе и IBOutlet и IBAction.
3. Перетянуть связь от необходимого UI - объекта до ViewController, и дальше выбрать из предложенных вариантов необходимое действие.
4. Из каких компонент состоит графический интерфейс программ на ObjectiveC?

**ОТВЕТ:**

ViewController.

App/SceneDelegates

Storyboard

Assets;

1. Что такое Outline View и Canvas в контексте InterfaceBuilder?

**ОТВЕТ:**

**Outline View** - представление, использующее формат строк и столбцов для отображения иерархических данных, таких как каталоги и файлы, которые можно разворачивать и сворачивать.

**Canvas** - тип вида, который поддерживает отрисовку в непосредственном режиме.

1. Что такое AutoLayout и Safe area?

**ОТВЕТ:**

**Auto Layout** занимается динамическим вычислением позиции и размера всех view в view иерархии, на основе constraints — правил заданных для того или иного view.

**Safe area** — это в основном предопределенная контрольная точка для верхней, нижней, левой (начальной) и правой (конечной) в Xcode

1. Что такое segue?

**ОТВЕТ:**

**Segue** — это отношение, установленное между двумя контроллерами представления.

1. Что такое StackView и constraints?

**ОТВЕТ:**

***constraints*** — правила заданные для того или иного *view*.

**Stack View** — это ваша палочка выручалочка при создании комплексных интерфейсов, если вы по каким-либо причинам не хотите использовать правила(*constraints*) или ваш интерфейс содержит в себе множество элементов расположение которых можно изменять бесконечно. Он может расставлять элементы внутри себя с данными параметрами.

1. В чем отличие между UIKit и SwiftUI?

**ОТВЕТ:**

[**UIKit**](https://developer.apple.com/documentation/uikit) — это императивный управляемый событиями фреймворк для создания пользовательских интерфейсов для платформы iOS. Это означает, что вы должны обрабатывать все изменения состояния во время таких событий, как загрузка представления, нажатие кнопки и т. д. Большим недостатком этого подхода является сложность синхронизации пользовательского интерфейса с его состоянием.

[**SwiftUI**](https://developer.apple.com/documentation/SwiftUI) — это декларативный фреймворк для создания пользовательских интерфейсов для всех платформ компании Apple. Ключевое слово в данном случае является декларативным. Декларативный означает, что разработчику нужно объявить, чего он хочет достичь, и фреймворк позаботится об этом.