



Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

---

ФАКУЛЬТЕТ ИУ «Информатика и системы управления»

КАФЕДРА ИУ-7 «Программное обеспечение эвм и информационные технологии»

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ  
НА ТЕМУ:**

***«Классификация методов верстки  
элементов интерфейса в разработке  
мобильных приложений под iOS»***

Студент      ИУ7-54Б

\_\_\_\_\_ Егорова П. А.

Руководитель

\_\_\_\_\_ Барышникова М. Ю.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

---

УТВЕРЖДАЮ

Заведующий кафедрой ИУ-7  
(Индекс)

И. В. Рудаков  
(И.О.Фамилия)

«16» сентября 2022 г.

**ЗАДАНИЕ**  
**на выполнение научно-исследовательской работы**

по теме

**«Классификация методов верстки элементов интерфейса  
в разработке мобильных приложений под iOS»**

Студент группы **ИУ7-54Б**

**Егорова Полина Александровна**

Направленность НИР

**учебная**

Источник тематики

**НИР кафедры**

График выполнения НИР: 25% к 6 нед., 50% к 9 нед., 75% к 12 нед., 100% к 15 нед.

***Техническое задание***

***Определить критерии, по которым осуществляется анализ эффективности использования методов верстки элементов интерфейса. Провести обзор существующих методов верстки мобильных приложений под iOS и выделить наиболее популярные. Провести классификацию методов на основе выделенных критериев.***

***Оформление научно-исследовательской работы:***

Расчетно-пояснительная записка на **12-20** листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т. п.)

Презентация на **6-10** слайдах.

Дата выдачи задания «16» сентября 2022 г.

Руководитель НИР

\_\_\_\_\_  
(Подпись, дата)

**М. Ю. Барышникова**  
(И.О.Фамилия)

Студент

\_\_\_\_\_  
(Подпись, дата)

**П. А. Егорова**  
(И.О.Фамилия)

# Оглавление

<b>Введение</b>	<b>3</b>
<b>1 Обзор предметной области</b>	<b>4</b>
1.1 Основные понятия . . . . .	4
1.2 UIKit . . . . .	5
<b>2 Методы верстки</b>	<b>7</b>
2.1 Ручная верстка . . . . .	7
2.1.1 Frame . . . . .	7
2.2 Автоматическая верстка . . . . .	9
2.2.1 Auto Layout . . . . .	9
2.2.2 Storyboard . . . . .	9
2.2.3 Xib . . . . .	9
<b>Список источников</b>	<b>10</b>

# Введение

iOS [1] – мобильная операционная система, разработанная и выпущенная компанией Apple [2] в 2007 году: инновационными в первых iPhone [3] стали возможность просмотра видео на YouTube [4] и поиск нескольких интернет-игр. За 15 лет система претерпела массу изменений: были разработаны новые технологии, создан язык программирования Swift [5], который пришел на смену первоначально использовавшемуся Objective-C [6], были выпущены гайдлайны [7], и всё это – дабы вырасти в многофункциональную платформу и превратить iPhone в некий «ПК в кармане». Это позволило Apple занять большое количество пользователей по всему миру, а, как следствие, и IT-специалистов, желающих создавать мобильные продукты под iOS.

Неотъемлемой частью iOS-разработки является создание интерфейса мобильного приложения. Каждый iOS-разработчик сталкивается с проблемой выбора метода разметки экранов продукта: Apple предоставляет несколько вариантов верстки. Возникает вопрос: по каким критериям осуществлять выбор и какая технология является в конкретном случае наиболее подходящей?

Целью данной работы является классификация методов верстки элементов интерфейса в разработке мобильных приложений под iOS. Для достижения поставленной цели необходимо решить следующие задачи:

- определить критерии, по которым осуществляется анализ эффективности использования методов верстки элементов интерфейса;
- провести обзор существующих методов верстки мобильных приложений под iOS и выделить наиболее популярные;
- классифицировать методы на основе выделенных критериев.

# 1 Обзор предметной области

Чтобы достичь намеченных целей, необходимо понять, что из себя представляет экран мобильного приложения, а также из каких элементов может состоять его интерфейс.

## 1.1 Основные понятия

В iOS существует различие между координатами, указываемыми в коде, и пикселями устройства [8] – наименьшими дискретными элементами двумерного цифрового изображения. Для большинства задач фактический размер точек [9] не имеет значения, их цель – обеспечить согласованный масштаб, который может использоваться в коде для указания размера и положения представлений и отображаемого содержимого. Например, если пиксели в два раза меньше изначальной высоты или ширины, можно использовать квадрат  $2 \times 2$  пикселя для каждой точки (это называется масштаб @2x). Таким образом, измерение в точках позволяет корректно масштабировать изображение на экранах с высоким разрешением. [7]

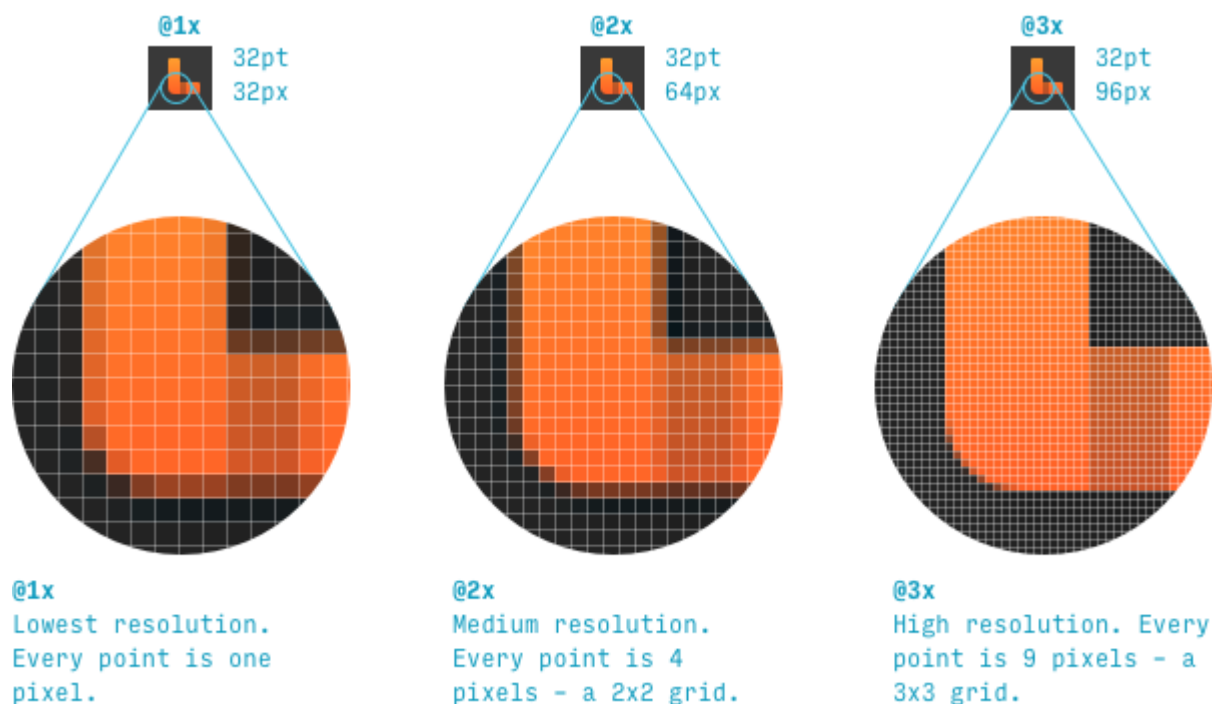


Рис. 1.1: Точки и пиксели

Устройства, работающие на операционной системе iOS, имеют различное разрешение экрана и соотношение сторон. Так, например, iPhone X имеет разрешение 1125 x 2436 пикселей и коэффициент масштабирования в точки, равный 3.0 (то есть 375 x 812 точек), а iPhone 6 – 750 x 1334 пикселей и коэффициент 2.0 соответственно (то есть 375 x 667) [10]. А также каждое устройство имеет горизонтальную и вертикальную ориентацию. Данные характеристики стоит учитывать в разработке, дабы создавать приложения, интерфейс которых адаптируется к устройствам разного размера, а также любой их ориентации.

## 1.2 UIKit

UIKit [11] – библиотека, предоставляющая архитектуру окон и представлений для реализации пользовательского интерфейса, включая компоненты, которые можно использовать для построения базовой инфраструктуры приложения. Альтернативой выступает фреймворк SwiftUI [12], предоставляющий не меньше возможностей, однако, будучи представленным в 2016 году, библиотека не смогла стать столь популярной в разработке. Поэтому целесообразным будет рассмотреть методы верстки UI-элементов, предоставляемых UIKit.

Рассмотрим базовые UI-элементы, предоставляемые фреймворком UIKit.

UIView (или представление) [13] – это фундаментальный блок пользовательского интерфейса приложения, а UIView-класс определяет поведение, общее для всех представлений. Объекты этого класса отображают содержимое в пределах своих границ и обрабатывают любые взаимодействия с этим содержимым. Для отображения надписей, изображений, кнопок и других элементов интерфейса, обычно встречающихся в приложениях, используют не определяемые самостоятельно подклассы view, а предоставляемые платформой UIKit.

UIViewController (или контроллер) [14] – объект, который управляет иерархией представлений приложения. Основные обязанности контроллера включают следующее:

- обновление содержимого представлений;

- реагирование на взаимодействие пользователя с представлениями;
- изменение размеров представлений и управление макетом общего интерфейса;
- координация с другими объектами, включая другие контроллеры представления.

Итак, экран в мобильной разработке представляет `UIViewController`, являющийся контейнером для других `UIView`. Труд команды разработки интерфейса мобильного приложения сводится к задаче корректного отображения элементов на экране – расположения `UIView` на `UIViewController`. Чтобы работа была эффективной, целесообразно каждому члену команды дать индивидуальную подзадачу – таким образом используемая технология верстки должна предполагать возможность работы нескольких участников над одним проектом и минимизировать количество ошибок, возникающих при изменении параметров UI-элементов.

На основе рассмотренных теоретических сведений можно выделить критерии, по которым необходимо провести анализ эффективности использования методов верстки:

- возможность создания интерфейса, масштабируемого для устройств разного размера и двух ориентаций экрана, и относительная сложность его создания;
- сложность командной разработки интерфейса; (сложность?)
- сложность внесения изменений в интерфейс;
- возможность обработки всех параметров UI-элемента;
- порог входа в технологию метода и ее наглядность; ???
- скорость работы метода.

Рассматривать разные элементы?

## 2 Методы верстки

Под понятием разметки подразумевается расчет необходимых координат и размеров UI-элементов. Существует два основных подхода к созданию пользовательского интерфейса: можно программно компоновать элементы, задавая координаты и размеры каждого в коде, или же использовать автоматическую компоновку.

### 2.1 Ручная верстка

#### 2.1.1 Frame

Frame [15] – свойство view, описывающее прямоугольник, определяющий местоположение и размер представления в системе координат его родительского view [16].

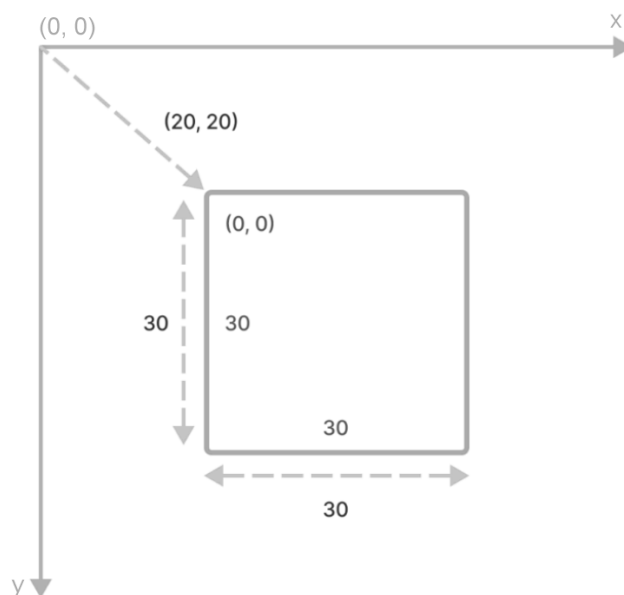


Рис. 2.1: Frame

View, представленную на рисунке 2.1, можно описать кодом, представленным в листинге 2.1.

```
1  var view = UIView()
2  view.frame = CGRect(x: 20, y: 20, width: 30, height: 30)
```

Листинг 2.1: Задание свойства frame для UIView



Ручная верстка предполагает задание кодом координат и размеров каждого элемента экрана. Указание координат идет относительно левого верхнего угла экрана – именно там располагается точка (0, 0). При желании запустить будущее приложение на двух устройствах, размеры экрана которых различны, результат также будет разниться. То же будет происходить и при попытке сравнения экранов горизонтальной и вертикальной ориентации. Таким образом, возникает необходимость вручную обрабатывать каждый из возможных размеров дисплея в двух вариациях ориентации. На данный момент существует  $n$  различных размеров экрана iPhone и  $m$  – iPad. Нетрудно представить, во сколько раз увеличится количество строк кода и повышется трудоемкость задания при обработке каждого случая.

Однако данный подход позволяет разбить задачу создания интерфейса на несколько подзадач, поскольку в пределах одной `Superview` разработчик взаимодействует лишь с этим родительским представлением и конкретной `view`. Внесение изменений в написанный ранее код также не составит большого труда, так как параметры, задаваемые в свойстве `frame`, независимы друг от друга и их значения являются константами.

При использовании данного метода разметки разработчик получает возможность создания любого кастомного элемента, поскольку имеет доступ к обработке каждого визуального параметра UI-элемента. Так, например, можно задавать радиус угла `view`, цвет и прозрачность элемента, что представлено в листинге 2.2.

```
1  var view = UIView()  
2  view.frame = CGRect(x: 20, y: 20, width: 30, height: 30)  
3  view.layer.cornerRadius = 2.0  
4  view.backgroundColor = UIColor.red.withAlphaComponent(0.5)
```

Листинг 2.2: Задание параметров `view`

## 2.2 Автоматическая верстка

### 2.2.1 Auto Layout

### 2.2.2 Storyboard

### 2.2.3 Xib

# Литература

- [1] iOS | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/packagedescription/platform/ios/>
- [2] Apple (Россия) – Официальный сайт / [Электронный ресурс]. Режим доступа: <https://www.apple.com/ru/>
- [3] iPhone – Apple (RU) – Официальный сайт / [Электронный ресурс]. Режим доступа: <https://www.apple.com/ru/iphone/>
- [4] YouTube / [Электронный ресурс]. Режим доступа: <https://www.youtube.com/>
- [5] The Swift Programming Language (Swift 5.7) / [Электронный ресурс]. Режим доступа: <https://www.apple.com/swift/>
- [6] About Objective-C / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html>
- [7] Human Interface Guidelines – Design – Apple Developer / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/design/human-interface-guidelines/guidelines/overview/>
- [8] Pixel | Definition and Meaning / [Электронный ресурс]. Режим доступа: <https://www.merriam-webster.com/dictionary/pixel#h1>
- [9] Point | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/uikit/uiaccessibilitylocationdescriptor/2890956-point?changes>
- [10] Displays | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/library/archive/documentation/DeviceInformation/Reference/iOSDeviceCompatibility/Displays/Displays.html>

- [11] UIKit | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/uikit/uikit>
- [12] SwiftUI | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/swiftui>
- [13] UIView | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/uikit/UIView>
- [14] UIViewController | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/uikit/uiviewcontroller>
- [15] Frame | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/uikit/UIView/1622621-frame>
- [16] Superview | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: <https://developer.apple.com/documentation/appkit/nsview/1483737-superview>