**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.**

*2022 г.*

**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.

# З А Д А Н И Е
## 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** _____ **М. Ю. Барышникова**
(Подпись, дата) (И.О.Фамилия)

**Студент** _____ **П. А. Егорова**
(Подпись, дата) (И.О.Фамилия)

# Оглавление

# Введение

iOS – мобильная операционная система [1], 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 заиметь 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 достижение 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 – наименьшими дискретными элементами двумерного цифрового изображения [9]. Для большинства задач фактический размер точек [10] 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 2×2 пикселя для каждой точки (это называется масштаб @2x). Таким образом, измерение в точках позволяет корректно масштабировать изображение на экранах с высоким разрешением. [7]

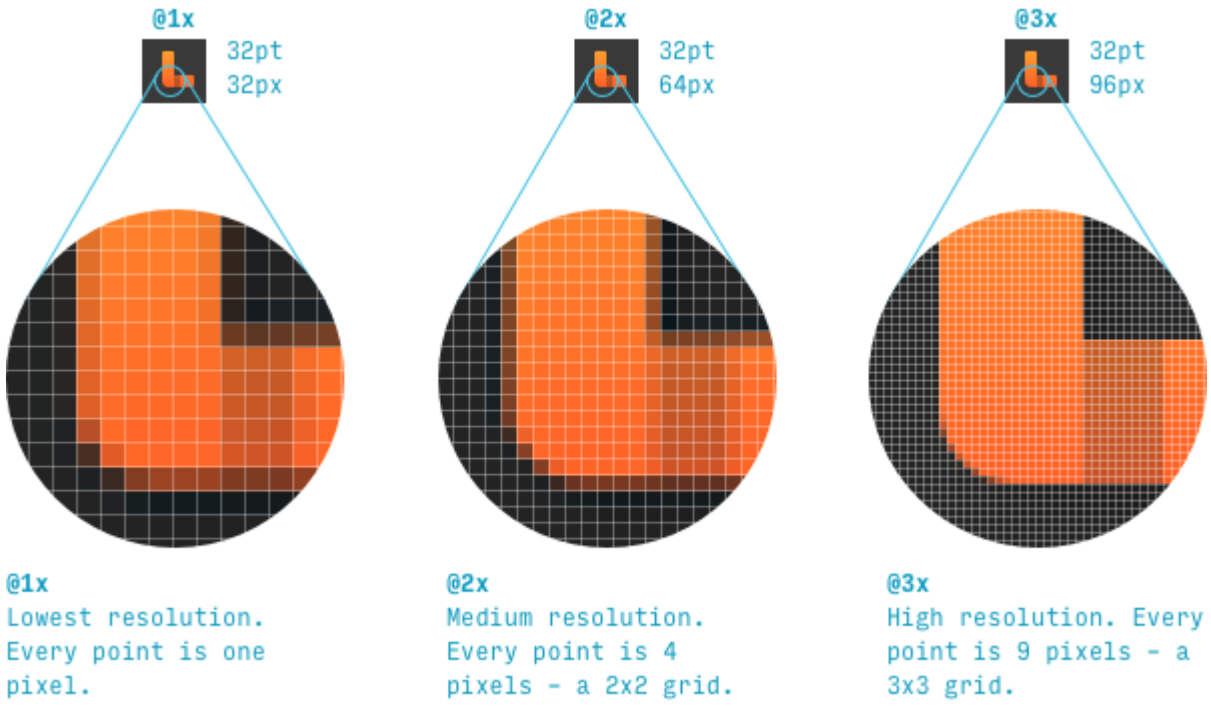


Рис. 1.1: Точки и пиксели

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). Данные характеристики стоит учитывать в разработке, дабы создавать приложения, интерфейс которых адаптируется к устройствам разного размера.

## 1.2 UIKit

UIKit – библиотека, предоставляющая архитектуру окон и представлений для реализации пользовательского интерфейса, включая компоненты, которые можно использовать для построения базовой инфраструктуры приложения [13]. Альтернативой выступает фреймворк SwiftUI [14], 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) – это фундаментальный блок пользовательского интерфейса приложения, а UIView-класс определяет поведение, общее для всех представлений [12]. 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) – объект, который управляет иерархией представлений приложения [15]. 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 масштабируемого для разных устройств интерфейса и относительная сложность его создания;
- относительная сложность командной разработки интерфейса;
- относительная сложность внесения изменений в интерфейс;
- порог входа в технологию метода и ее наглядность; ???
- возможность обработки всех параметров UI-элемента;
- скорость работы метода.

Рассматривать разные элементы?

# 2 Методы верстки

Существует три основных подхода к созданию пользовательского интерфейса: можно программно компоновать пользовательский интерфейс с помощью, использовать маски автоматического изменения размера, чтобы автоматизировать некоторые реакции на внешние изменения, или использовать автоматическую компоновку.

## 2.1 Ручная верстка

### 2.1.1 Frame

## 2.2 Автоматическая верстка

### 2.2.1 Storyboard

### 2.2.2 Xib

### 2.2.3 Auto Layout

# Заключение

В ходе выполнения лабораторной работы были решены следующие задачи:

- были изучены и реализованы 3 алгоритма сортировки массивов: блочный, перемешиванием, бинарным деревом;
- был произведен анализ трудоёмкости алгоритмов на основе теоретических расчетов и выбранной модели вычислений;
- оценена эффективность алгоритмов по времени на основе экспериментальных данных: выявлено, что сортировка перемешиванием проигрывает по памяти конкурентным реализациям при любом наборе входных данных: отсортированном массива, неотсортированном и массиве случайных чисел. Самым быстрым оказался блочный алгоритм, хоть данная реализация и незначительно медленнее сортировки бинарным деревом;
- оценена эффективность алгоритмов по объему занимаемой памяти: выявлено, что сортировка бинарным деревом проигрывает по этому параметру двум другим в виду использования дополнительной памяти для хранения структуры бинарного дерева. Сортировка перемешиванием и блочная не имеют больших различий по рассматриваемому параметру.

# 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] frame | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/uikit/uiview/1622621-frame

[9] Pixel | Definition and Meaning / [Электронный ресурс]. Режим доступа: https://www.merriam-webster.com/dictionary/pixel#h1

[10] point | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/uikit/uiaccessibilitylocationdescriptor/2890956-point?changes

[11] point | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://design.sredaobuchenia.ru/ios

[12] UIView | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/uikit/uiview

[13] UIKit | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/uikit/uikit

[14] SwiftUI | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/swiftui

[15] UIViewController | Apple Developer Documentation / [Электронный ресурс]. Режим доступа: https://developer.apple.com/documentation/uikit/uiviewcontroller