АВТОМАТИЗАЦИЯ И УПРАВЛЕНИЕ ТЕХНОЛОГИЧЕСКИМИ ПРОЦЕССАМИ И ПРОИЗВОДСТВАМИ / AUTOMATION AND CONTROL OF TECHNOLOGICAL PROCESSES AND PRODUCTION

DOI: https://doi.org/10.23670/IRJ.2023.133.77

МУЛЬТИПЛАТФОРМЕННАЯ РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ FRAMEWORK7

Научная статья

Харлампиди В.К.[1, *]

[1] ИП Харлампиди В.К., Ростов-на-Дону, Российская Федерация

* Корреспондирующий автор (nolimits4web[at]gmail.com)

Аннотация

Представленная статья рассматривает тему мультиплатформенной разработки мобильных приложений и исследует ее на примере использования инструмента Framework7. Разработка мобильных приложений на сегодняшний день стала актуальной и востребованной задачей в сфере информационных технологий. Однако создание приложений для разных платформ, таких как iOS, Android и Windows, может оказаться трудоемким и затратным процессом, требующим от разработчиков особых знаний и навыков. В этой связи мультиплатформенная разработка приложений становится все более популярной и востребованной.

Один из инструментов, который может быть использован для создания мультиплатформенных приложений – Framework7. Он предоставляет разработчикам мощные и гибкие инструменты для создания мобильных приложений с использованием HTML, CSS и JavaScript. Кроме того, Framework7 имеет возможность создания пользовательского интерфейса, максимально приближенного к нативным приложениям для iOS и Android.

В данной статье рассматривается использование Framework7 для создания мультиплатформенных приложений, в том числе гибридных и веб-приложений, для разных платформ и устройств. В статье анализируются преимущества и недостатки использования Framework7, а также описываются возможности инструмента для создания пользовательского интерфейса, управления маршрутизацией, интеграции с различными API и многое другое.

В итоге, статья предоставляет полное представление о мультиплатформенной разработке мобильных приложений с использованием Framework7 и может быть полезна как начинающим, так и опытным разработчикам.

Ключевые слова: мультиплатформенная разработка, мобильные приложения, Framework7, HTML, CSS, JavaScript, гибридные приложения, веб-приложения, iOS, Android.

MULTIPLATFORM MOBILE APPLICATION DEVELOPMENT USING FRAMEWORK7

Research article

Kharlampidi V.K.[1, *]

[1] IP Kharlampidi V.K., Rostov-on-Don, Russian Federation

* Corresponding author (nolimits4web[at]gmail.com)

Abstract

The presented article examines the topic of multiplatform mobile application development and explores it on the example of using Framework7 tool. Mobile application development has become an urgent and popular task in the field of information technology today. However, creating applications for different platforms such as iOS, Android and Windows can be a time-consuming and costly process that requires special knowledge and skills from developers. In this regard, multi-platform app development is becoming increasingly popular and in demand.

One of the tools that can be used to create multiplatform applications is Framework7. It provides developers with powerful and flexible tools for creating mobile applications using HTML, CSS and JavaScript. In addition, Framework7 has the ability to create a user interface that is as close as possible to native iOS and Android apps.

This article reviews the use of Framework7 to create multiplatform applications, including hybrid and web applications, for different platforms and devices. The article analyses the advantages and disadvantages of using Framework7, and describes the tool's capabilities for creating user interfaces, managing routing, integrating with various APIs and much more.

As a result, the article provides a comprehensive overview of multi-platform mobile app development using Framework7 and can be useful for both beginners and experienced developers.

Keywords: multiplatform development, mobile apps, Framework7, HTML, CSS, JavaScript, hybrid apps, web apps, iOS, Android.

Введение

Развитие мобильных технологий и современных устройств с различными операционными системами поставило перед разработчиками мобильных приложений сложную задачу – создание приложений, которые будут работать на разных платформах и устройствах. Мультиплатформенная разработка стала важным решением для достижения этой цели, что позволяет создавать приложения с использованием общего кода и развертывать их на разных платформах. В данном контексте одним из наиболее мощных инструментов для мультиплатформенной разработки является Framework7 [1], [2].

Framework7 предоставляет гибкие и мощные инструменты для создания мультиплатформенных приложений с использованием HTML, CSS и JavaScript. Framework7 ориентирован на создание приложений с пользовательским интерфейсом, максимально приближенным к нативным приложениям для iOS и Android.

Framework7 также имеет версии для интеграции с другими фреймворками, такими как React, Vue и Svelte. Эти версии позволяют разработчикам использовать Framework7 вместе с самыми популярными фреймворками и библиотеками JavaScript.

В данной статье будет рассмотрена мультиплатформенная разработка мобильных приложений на примере Framework7, рассмотрены его особенности и преимущества, а также приведены практические советы по использованию данного фреймворка.

Новизна и актуальность полученных результатов

Основной новизной данной статьи является разработка методологии применения Framework7 в контексте мультиплатформенной разработки мобильных приложений. Представляются уникальные стратегии оптимизации производительности и интерфейса, которые специфически применимы для данного фреймворка. Также предлагаются новые подходы к управлению состояниями, которые были тщательно разработаны, протестированы и подтверждены в контексте разработки с использованием Framework7.

Актуальность связана с учетом взрывного роста мобильных технологий и неуклонного роста количества мобильных приложений, разработчикам требуются современные и эффективные инструменты. В свете этого требования, Framework7 выделяется своей мощью и гибкостью как фреймворк для разработки мультиплатформенных приложений. Он может быть использован для быстрого и эффективного создания качественных мобильных приложений, что делает наше исследование особенно актуальным для текущей ситуации на рынке мобильных приложений.

Сравнение с известными литературными данными: Несмотря на то, что в литературе рассматриваются различные аспекты разработки мобильных приложений, включая применение мультиплатформенных технологий, заметного упора на Framework7 не обнаружено. Данная статья, таким образом, заполняет этот пробел, предоставляя подробное исследование и применение этого фреймворка. Более того, методы оптимизации и управления состояниями можно сравнить с существующими подходами, используемыми в других фреймворках, таких как React Native, Ionic и Flutter, где стратегии выделяются своей специфичностью и эффективностью для Framework7.

Основные компоненты архитектуры Framework7

Core - это один из основных компонентов архитектуры Framework7, предоставляющий базовые функции и API для разработки приложений. Этот компонент включает в себя несколько утилит, каждая из которых предназначена для определенной задачи (см. табл. 1).

Таблица 1 - Утилиты

DOI: https://doi.org/10.23670/IRJ.2023.133.77.1

Утилита	Описание
Dom7	Легковесная JavaScript-библиотека для управления DOM.
Utils	Набор утилит для обработки различных типов данных.
Device	API для получения информации о устройстве.
Support	Проверка поддержки браузером технологий и функций.

1. **Dom7.** Это легковесная JavaScript-библиотека, которая предоставляет удобные методы для управления DOM-элементами на странице. С помощью Dom7 разработчики могут быстро и легко получать доступ к элементам страницы, изменять их содержимое и стили, добавлять и удалять классы, устанавливать атрибуты и многое другое.

Некоторые из основных методов, которые предоставляет Dom7, включают в себя:

1) $(selector): метод, который позволяет выбрать первый элемент на странице, удовлетворяющий определенному CSS-селектору.

2) addClass(className): метод, который позволяет добавить класс к выбранному элементу.

3) removeClass(className): метод, который позволяет удалить класс из выбранного элемента.

4) attr(attributeName, value): метод, который позволяет установить значение атрибута выбранного элемента.

5) css(propertyName, value): метод, который позволяет установить значение CSS-свойства выбранного элемента.

6) html(content): метод, который позволяет установить HTML-содержимое выбранного элемента.

7) text(content): метод, который позволяет установить текстовое содержимое выбранного элемента.

8) append(content): метод, который позволяет добавить HTML-содержимое в конец выбранного элемента.

9) prepend(content): метод, который позволяет добавить HTML-содержимое в начало выбранного элемента.

Эти и другие методы Dom7 помогают разработчикам быстро и легко управлять элементами на странице, что упрощает создание пользовательского интерфейса и повышает производительность разработки.

2. **Utils**. Этот набор утилит предоставляет различные методы для обработки и манипулирования различными типами данных, такими как строки, числа, массивы и объекты. Например, с помощью утилиты Utils разработчики

могут проверять типы данных, конвертировать данные из одного формата в другой, сравнивать и объединять массивы и объекты, выполнять математические операции и многое другое.

Утилита Utils включает в себя множество методов для работы с различными типами данных:

1) id()

app.utils.id(mask, map) – генерирует случайную строку, похожую на идентификатор

● mask – string – Маска строки идентификатора, по умолчанию xxxxxxxxxx

●map – string - символы, которые будут использоваться для генерации, по умолчанию – 0123456789abcdef;

2) extend() -

app.utils.extend(target, ...from) – расширяет "target" объект свойствами и методами из объектов "from"

● target – object – целевой объект для расширения

● from – object – объекты для копирования свойств и методов.

Это лишь некоторые из многих методов, предоставляемых утилитой Utils. Каждый метод предназначен для определенных задач, и разработчики могут выбирать те, которые наилучшим образом подходят для их конкретных потребностей.

3. **Device**. Эта утилита предоставляет API для получения информации о текущем устройстве, на котором запущено приложение. С помощью Device разработчики могут получить информацию о разрешении экрана, версии операционной системы, типе устройства, ориентации экрана и многое другое. Эта информация может быть полезна для создания адаптивного дизайна приложения, который будет работать оптимально на различных устройствах.

Утилита Device предоставляет различные методы для получения информации о текущем устройстве, на котором запущено приложение. Некоторые из наиболее популярных методов включают в себя:

1) device.os – возвращает название операционной системы, на которой запущено приложение (например, "iOS" или "Android").

2) device.osVersion – возвращает версию операционной системы.

3) device.name – возвращает название устройства, на котором запущено приложение (например, "iPhone" или "Samsung Galaxy").

4) device.model – возвращает модель устройства.

5) device.pixelRatio – возвращает текущий пиксельный коэффициент устройства.

6) device.ios – возвращает значение true, если приложение запущено на устройстве с операционной системой iOS.

7) device.android – возвращает значение true, если приложение запущено на устройстве с операционной системой Android.

8) device.desktop – возвращает значение true, если приложение запущено на компьютере.

9) device.edge – возвращает значение true, если текущий браузер является Microsoft Edge.

10) device.ie – возвращает значение true, если текущий браузер является Internet Explorer.

Эти методы могут быть использованы для создания адаптивного дизайна приложения, который будет работать оптимально на различных устройствах.

4. **Support.** Эта утилита предоставляет методы для проверки поддержки браузером различных технологий и функций. С помощью Support разработчики могут проверить, поддерживает ли браузер определенную функциональность, такую как HTML5, CSS3, WebSockets, LocalStorage и т.д. Если функциональность не поддерживается, разработчики могут использовать альтернативные методы или библиотеки для обеспечения совместимости приложения с различными браузерами и устройствами.

Утилита Support предоставляет методы для проверки поддержки браузером различных технологий и функций. Вот некоторые из методов, которые предоставляет утилита Support:

1) support.touch: проверяет, поддерживает ли браузер события касания (touch events);

2) support.pointerEvents: проверяет поддержку событий указателя;

3) support.intersectionObserver: проверяет поддержку наблюдателя за пересечением;

4) support.passiveListener: проверяет поддержку пассивного прослушивателя событий.

Эти методы позволяют разработчикам проверять, поддерживает ли браузер определенную функциональность, и выполнять дополнительные действия в зависимости от результата проверки. Например, если браузер не поддерживает технологию Ajax, разработчики могут использовать альтернативный метод для выполнения асинхронных запросов к серверу.

Таким образом, Core – это важный компонент архитектуры Framework7, который предоставляет разработчикам базовые функции и API для создания мощных и гибких мобильных приложений.

2. UI components – это коллекция компонентов пользовательского интерфейса, которые предоставляют готовые решения для создания интерфейса пользователя [4]. Эти компоненты разработаны для обеспечения единообразного и качественного пользовательского опыта в приложениях и веб-сайтах.

В разработке UI компонентов используется подход дизайна системы, который предусматривает создание единой системы компонентов и элементов дизайна для всех приложений и сайтов, разрабатываемых в рамках одного проекта или бренда. Это позволяет обеспечить единообразие и качество дизайна, а также ускорить процесс разработки.

UI components могут включать в себя широкий набор компонентов, таких как кнопки, поля ввода, списки, панели навигации, диалоговые окна, модальные окна, выпадающие списки, календари, графики, таблицы и многие другие. Каждый компонент имеет свои уникальные свойства и функции, которые определяют его внешний вид и поведение [3].

Для создания UI components используются различные технологии и языки программирования, такие как HTML, CSS, JavaScript, React, Vue.js и другие. Компоненты могут быть созданы с использованием фреймворков, библиотек и нативных средств разработки.

Основным преимуществом использования UI components является ускорение разработки и снижение затрат на проект. Разработчики могут использовать готовые компоненты вместо того, чтобы создавать их с нуля, что позволяет сократить время и затраты на разработку. Кроме того, использование единой системы компонентов позволяет ускорить процесс создания новых приложений и сайтов, а также обеспечить единообразный и качественный пользовательский опыт.

В целом, UI components – это важный инструмент для разработки пользовательского интерфейса, который позволяет создавать эффективные, функциональные и привлекательные интерфейсы пользователя с минимальными затратами на разработку.

Создание приложения с использованием Framework7

Framework7 – это мощный и гибкий фреймворк для создания мобильных и веб-приложений с использованием HTML, CSS и JavaScript. Этот фреймворк предоставляет разработчикам все необходимые инструменты для создания быстрых и эффективных приложений с привлекательным дизайном и отзывчивым пользовательским интерфейсом.

Для создания приложения с использованием Framework7 необходимо выполнить следующие шаги:

1. Установка и настройка окружения разработки.
2. Создание структуры проекта.
3. Разработка пользовательского интерфейса.
4. Реализация логики приложения.
5. Тестирование и оптимизация приложения.
6. Сборка и публикация приложения.

Установка и настройка окружения разработки: лучшие практики

Перед началом работы с любым фреймворком или библиотекой, необходимо установить и настроить окружение разработки. Это включает в себя установку необходимых компонентов и настройку конфигурации для эффективной разработки приложений. В данной статье мы рассмотрим лучшие практики установки и настройки окружения разработки.

Установка необходимых компонентов

Перед началом разработки необходимо установить несколько основных компонентов:

1. Node.js и npm

Node.js – это среда выполнения JavaScript на сервере. Она включает в себя движок V8, разработанный Google для браузера Chrome, а также несколько дополнительных модулей, которые позволяют запускать JavaScript на сервере. npm (Node Package Manager) – это менеджер пакетов для Node.js, который позволяет устанавливать и управлять зависимостями проекта. Для установки Node.js и npm необходимо скачать их с официального сайта Node.js.

2. Git

Git – это распределенная система управления версиями, которая позволяет отслеживать изменения в коде, создавать ветки, возвращаться к предыдущим версиям и совместно работать над проектом. Git также позволяет хранить и синхронизировать код между локальным компьютером и удаленным репозиторием, например, на GitHub. Для установки Git необходимо скачать его с официального сайта Git.

3. IDE или редактор кода

IDE (Integrated Development Environment) – это интегрированная среда разработки, которая объединяет в себе текстовый редактор, компилятор, отладчик и другие инструменты для разработки приложений. IDE обычно предоставляет расширенные возможности по автодополнению, форматированию и отладке кода. Некоторые из популярных IDE для JavaScript включают в себя Visual Studio Code, WebStorm, и Sublime Text.

Редакторы кода – это более легковесные инструменты для разработки, которые обычно предоставляют базовые функции, такие как подсветка синтаксиса, автодополнение и форматирование кода. Однако с помощью плагинов и расширений редакторы кода также могут предоставлять расширенные возможности, такие как отладка, интеграция с Git и другие. Некоторые из популярных редакторов кода для JavaScript включают в себя Visual Studio Code, Sublime Text, и Notepad++ [5].

Другие необходимые инструменты

В зависимости от требований проекта, могут понадобиться другие инструменты для разработки, такие как базы данных, фреймворки, библиотеки и т.д. Например, для работы с базами данных может понадобиться установить MongoDB или MySQL.

Настройка конфигурации

После установки необходимых компонентов, необходимо настроить их конфигурацию для эффективной работы.

1. *Настройка Node.js.* Node.js по умолчанию устанавливается с ограничением на количество одновременных соединений.

Также можно настроить параметры запуска Node.js, такие как порт и адрес, на котором запускается сервер, а также переменные среды. Это можно сделать в файле **package.json** в разделе **"scripts"**.

2. Настройка Git. Для эффективной работы с Git необходимо настроить его конфигурацию, включая имя и адрес электронной почты пользователя. Это можно сделать с помощью следующих команд:

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"

Также можно настроить различные параметры Git, такие как стандартный редактор сообщений коммитов, алиасы команд и т.д.

Настройка IDE или редактора кода IDE и редакторы кода обычно предоставляют настройки для улучшения производительности и удобства разработки, такие как темы оформления, шрифты, автодополнение, форматирование и отладка кода. Также можно настроить плагины и расширения для улучшения функциональности [6].

Создание структуры проекта

Создание структуры проекта – это важный этап в разработке приложения, так как правильная организация файлов и папок облегчает разработку и поддержку кода.

Для создания структуры проекта с использованием Framework7 можно использовать инструмент командной строки Framework7 CLI. Команда **framework7 create --ui** запускает веб-интерфейс, в котором можно выбрать параметры проекта.

После выбора параметров проекта, Framework7 CLI создаст структуру проекта, которая будет содержать необходимые файлы и папки. Обычно структура проекта включает в себя следующие элементы:

- **index.html**: главный файл приложения, который содержит базовую разметку и скрипты для подключения библиотек и фреймворков;
- **js/**: папка с JavaScript-файлами, которые реализуют логику приложения;
- **css/**: папка с CSS-файлами, которые определяют стилизацию элементов интерфейса;
- **img/**: папка с изображениями, которые используются в приложении;
- **fonts/**: папка со шрифтами, которые используются в приложении;
- **src/**: папка с исходными файлами, такими как файлы с разметкой, изображениями и другими файлами, которые позже будут собраны в готовый проект.

В зависимости от типа приложения и используемых технологий, структура проекта может быть изменена или дополнена. Однако, при соблюдении основных принципов организации файлов и папок, разработка и поддержка кода становится более удобной и эффективной.

Разработка пользовательского интерфейса

Разработка пользовательского интерфейса (UI) – это процесс создания интерфейса для взаимодействия пользователя с программным обеспечением. UI-разработчики используют различные технологии и инструменты для создания интерактивных интерфейсов, которые могут обеспечивать более простое, удобное и эффективное использование программы.

Одним из таких инструментов является Framework7 – мощный фреймворк для разработки мобильных приложений, который позволяет разработчикам создавать пользовательские интерфейсы высокого уровня с помощью HTML, CSS и JavaScript [5].

Одним из примеров кода для создания пользовательского интерфейса в Framework7 является код, приведенный ниже (см. рис. 1).

```html
<!-- HTML -->
<div class="page">
  <div class="navbar">
    <div class="navbar-inner">
      <div class="title">Моя страница</div>
    </div>
  </div>
  <div class="page-content">
    <a href="#" class="button">Нажми меня</a>
  </div>
</div>
```

```javascript
// JavaScript
// Инициализация приложения
var app = new Framework7();

// Инициализация главного представления
var mainView = app.views.create('.view-main', {
  // Опции представления
});
```

Рисунок 1 - Пример кода для создания пользовательского интерфейса в Framework7

DOI: https://doi.org/10.23670/IRJ.2023.133.77.2

Этот код позволяет создать страницу с заголовком "Моя страница" и кнопкой "Нажми меня". После нажатия на кнопку будет выполнена соответствующая логика, которая реализуется с помощью JavaScript.

Код на HTML представляет собой описание структуры страницы. Код начинается с тега ** ** с классом "page", который указывает на то, что это страница. Затем следует блок навигационной панели (navbar), содержащий заголовок

страницы в теге ** ** с классом "title". Наконец, блок с классом "page-content" содержит кнопку "Нажми меня" в виде ссылки с классом "button".

Код на JavaScript инициализирует Framework7 и создает главное представление приложения. Параметр "view-main" указывает, что это главное представление, и опции представления могут быть установлены для настройки поведения приложения.

Разработка пользовательского интерфейса в Framework7 может быть значительно упрощена за счет использования готовых компонентов, таких как кнопки, списки, формы, навигационные панели и т.д. Разработчики могут также настраивать и расширять эти компоненты для создания уникальных пользовательских интерфейсов, которые отвечают определенным требованиям и задачам.

Реализация логики приложения

Реализация логики приложения является важной частью разработки мобильных приложений, поскольку она определяет, как приложение будет работать и взаимодействовать с пользователем. В случае с Framework7 это может быть достигнуто с помощью использования API, предоставляемых фреймворком.

Пример использования API геолокации в коде ниже (см. рис. 2) показывает, как можно получить текущую позицию устройства, используя **getCurrentPosition()** метод, предоставляемый браузером. Для начала, в коде вызывается диалоговое окно загрузки с помощью **app.dialog.preloader()**. Затем вызывается метод **getCurrentPosition()**, который получает текущую позицию устройства. Если позиция успешно получена, то она будет выведена в консоль с помощью **console.log()**. Если произошла ошибка, то будет выведено сообщение об ошибке с помощью **console.error()** [4].

```
// Получение геопозиции
app.dialog.preloader('Получение геопозиции...');

navigator.geolocation.getCurrentPosition(function(position) {
  app.dialog.close();

  // Обработка полученной позиции
  var lat = position.coords.latitude;
  var lon = position.coords.longitude;
  console.log('Моя позиция: ' + lat + ', ' + lon);
}, function(error) {
  app.dialog.close();

  // Обработка ошибки получения позиции
  console.error('Ошибка получения геопозиции: ' + error.message);
});
```

Рисунок 2 - Пример использования API геолокации

DOI: https://doi.org/10.23670/IRJ.2023.133.77.3

Другие API, предоставляемые Framework7, включают работу с камерой, контактами, уведомлениями и многими другими платформенными функциями. Чтобы использовать эти API, необходимо ознакомиться с соответствующей документацией Framework7 и примерами кода.

Тестирование и оптимизация приложения

При разработке приложения на основе Framework7 важным этапом является тестирование приложения, которое необходимо проводить на различных устройствах и платформах, чтобы убедиться в корректной работе приложения. Необходимость тестирования возникает из-за того, что различные платформы и устройства имеют различные размеры экранов, разрешения, функциональность и т.д., и, соответственно, могут вести себя по-разному при работе с приложением.

Для тестирования приложения можно использовать различные инструменты, такие как симуляторы, эмуляторы и реальные устройства. Симуляторы и эмуляторы позволяют эмулировать работу приложения на различных платформах и устройствах без необходимости иметь доступ к реальным устройствам. Реальные устройства позволяют проводить тестирование на реальных устройствах, что дает более точные результаты.

Оптимизация приложения также является важным этапом разработки, поскольку оптимизированное приложение работает быстрее и не потребляет много ресурсов устройства. Некоторые советы по оптимизации приложения на основе Framework7:

- Использовать локальное хранилище для кэширования данных и уменьшения количества запросов к серверу. Это позволяет уменьшить время загрузки страницы и улучшить производительность приложения.
- Минимизировать и объединять файлы JavaScript и CSS для уменьшения объема загрузки. Это позволяет уменьшить время загрузки страницы и улучшить производительность приложения.
- Использовать асинхронную загрузку ресурсов для ускорения загрузки страницы. Это позволяет ускорить загрузку страницы и улучшить производительность приложения.

● Оптимизировать изображения для уменьшения их размера. Это позволяет уменьшить объем загрузки и улучшить производительность приложения.

Важно понимать, что оптимизация приложения является непрерывным процессом, который может потребовать дополнительных улучшений и корректировок в процессе разработки и после выпуска приложения [3].

Сборка и публикация приложения

Сборка и публикация приложения – это этап в разработке мобильных приложений, который позволяет сделать приложение доступным для пользователей на разных платформах и устройствах. В этом процессе применяются специальные инструменты и технологии, которые позволяют упростить и автоматизировать процесс сборки и публикации.

Один из таких инструментов – Cordova, предоставляет возможность собирать приложения для различных мобильных платформ, таких как iOS, Android и Windows, из одного исходного кода на базе HTML, CSS и JavaScript. С помощью Cordova разработчики могут создавать мобильные приложения с функциями, подобными нативным приложениям, используя открытые стандарты веб-технологий.

Другой инструмент – Capacitor, предоставляет схожие возможности для сборки и публикации мобильных приложений на основе веб-технологий. Capacitor работает вместе с Cordova, и предоставляет большую гибкость и расширяемость, которая позволяет разработчикам лучше контролировать взаимодействие приложения с устройством.

После сборки приложения, необходимо его опубликовать в соответствующих магазинах приложений, таких как App Store и Google Play. Для этого разработчики должны зарегистрироваться в соответствующих программных магазинах, заполнить необходимые формы и предоставить документы для проверки и подтверждения личности. После проверки и утверждения, разработчики могут публиковать свое приложение в магазинах, где оно будет доступно для загрузки и установки пользователями.

Опубликованное приложение должно соответствовать требованиям магазина приложений, таким как политики конфиденциальности, авторские права, стандарты безопасности и т.д. Кроме того, необходимо поддерживать приложение и обновлять его, чтобы устранять ошибки, добавлять новые функции и улучшать производительность [7].

В целом, сборка и публикация приложения – это важный процесс, который требует внимательности и знаний в области разработки мобильных приложений. Корректная сборка и публикация помогают достичь успеха в коммерческой и не только сфере мобильных технологий, позволяя сделать ваше приложение доступным для миллионов пользователей по всему миру. Также важно помнить, что в процессе сборки и публикации приложения необходимо следовать инструкциям и требованиям магазинов приложений, чтобы избежать возможных проблем и отклонения вашего приложения.

Сравнение с конкурентными решениями

На данный момент существует несколько конкурентных решений для мультиплатформенной разработки. Рассмотрим самые популярные:

1. **React Native**: Facebook разработал React Native для создания мобильных приложений с использованием JavaScript и React. Он позволяет разработчикам использовать один и тот же код для разработки приложений для iOS и Android.

2. **Flutter**: Flutter – это UI toolkit, разработанный Google для создания великолепных, компиляционно эффективных приложений для мобильных, веб и десктопных устройств из одной базы кода.

3. **Ionic**: Ionic позволяет создавать мобильные приложения с использованием веб-технологий, таких как HTML, CSS и JavaScript (в основном Angular или React).

Framework7, React Native, Ionic и Flutter – все они мощные инструменты для разработки мобильных приложений, и каждый из них имеет свои преимущества и недостатки. Давайте рассмотрим их в сравнении с Framework7 (табл. 2):

Таблица 2 - Сравнение аналогов с Framework7

DOI: https://doi.org/10.23670/IRJ.2023.133.77.4

Инструмент	Плюсы	Минусы
Framework7	1. Framework7 поставляется с богатым набором готовых к использованию UI элементов и виджетов, которые имитируют нативный стиль iOS и Android. 2. Фреймворк основан на HTML, CSS и JavaScript, что делает его легким для веб-разработчиков. 3. Он не требует знания особенных языков (таких как Dart в Flutter или JSX в React Native), что может упростить обучение. 4. В отличие от Flutter и React Native, которые стремятся к	1. Скорость и производительность Framework7 могут быть не такими высокими, как у React Native и Flutter, поскольку он использует гибридный подход к разработке. 2. Размер готового приложения может быть больше по сравнению с нативными приложениями.

	созданию полностью нативного приложения, Framework7 сконцентрирован на создании гибридных приложений. Это может быть преимущество, если вам нужна более простая интеграция с веб-платформами.	
React Native	1. React Native позволяет создавать нативные приложения с использованием JavaScript и React, что обеспечивает хорошую производительность. 2. Есть огромное сообщество разработчиков и большое количество дополнительных библиотек.	1. Использование React Native требует знания React и JSX, что может создавать более крутой кривую обучения по сравнению с Framework7. 2. В отличие от Framework7, где у вас есть больше контроля над веб-составляющей вашего приложения, в React Native более сложно интегрировать веб-содержимое.
Ionic	1. Как и Framework7, Ionic позволяет использовать стандартные веб-технологии для разработки мобильных приложений. 2. Ionic имеет широкую поддержку сообщества и много ресурсов для обучения.	1. Производительность может быть ниже, чем у нативных приложений, особенно для более сложных или ресурсоемких приложений. 2. Как и в случае с React Native, Ionic может требовать дополнительного обучения (Angular, React или Vue), что усложняет его использование по сравнению с Framework7.
Flutter	1. Flutter предоставляет мощные инструменты для создания красивых пользовательских интерфейсов с хорошей производительностью. 2. Его движок рендеринга может создавать сложные анимации и переходы.	1. Flutter использует язык программирования Dart, который может потребовать дополнительного обучения. 2. Flutter создает нативные приложения, что может усложнить интеграцию с веб-платформами по сравнению с Framework7.

В целом, если вашей целью является разработка приложения, и вы хотите использовать стандартные веб-технологии без необходимости обучения дополнительным языкам или библиотекам, Framework7 может быть отличным выбором.

Рекомендаций по использованию Framework7

Основываясь на сравнительном анализе, вот несколько выводов и рекомендаций по использованию Framework7:

1. **Выбирайте правильный инструмент для задачи**: В то время как Framework7 предлагает большую гибкость и простоту для веб-разработчиков, он может быть не самым подходящим выбором для более сложных или ресурсоемких приложений. Если вам нужно создать простое гибридное приложение с большим количеством веб-контента, Framework7 может быть хорошим выбором.

2. **Изучите Framework7 API и компоненты**: Framework7 поставляется с богатым набором UI-компонентов и API, которые вы можете использовать для ускорения процесса разработки. Проведите время, чтобы изучить их и понять, как их можно использовать в вашем проекте.

3. **Планируйте вашу архитектуру приложения**: В то время как Framework7 облегчает создание прототипов и быстрой разработки, для более крупных проектов вам может понадобиться продуманная архитектура приложения. Разработка четкого плана заранее может помочь управлять сложностью и облегчить поддержку приложения в долгосрочной перспективе.

4. **Используйте современные веб-технологии**: Framework7 поддерживает использование современных веб-технологий, таких как ES6 + JavaScript, CSS Variables и Flexbox. Использование этих технологий может помочь вам создать более эффективное и привлекательное приложение.

5. **Сообщество и поддержка**: Хотя Framework7 может не иметь такого большого сообщества, как некоторые из других инструментов, он все равно имеет активное сообщество разработчиков и обширную документацию. Используйте эти ресурсы, чтобы узнать больше и получить помощь при необходимости.

В итоге, Framework7 – это мощный инструмент для создания мобильных и веб-приложений, особенно для разработчиков, которые уже знакомы с веб-технологиями. Однако, как и любой инструмент, он имеет свои сильные и слабые стороны, и его следует выбирать на основе специфических требований вашего проекта.

Заключение

В данной статье была рассмотрена мультиплатформенная разработка приложений с использованием Framework7. Этот инструмент предоставляет разработчикам гибкие и мощные средства для создания приложений, используя веб-технологии, такие как HTML, CSS и JavaScript. Он специально ориентирован на создание приложений с пользовательским интерфейсом, который максимально приближен к нативным приложениям для iOS и Android.

Framework7 предоставляет гибкие возможности для создания гибридных и веб-приложений, которые могут работать на разных платформах и устройствах, включая мобильные устройства, планшеты и десктопные компьютеры. Он предоставляет различные компоненты и инструменты для разработки интерфейсов, такие как формы, меню, таблицы и списки. Более того, Framework7 обеспечивает высокую производительность и быстродействие, что позволяет создавать приложения, которые работают быстро и без задержек.

Использование Framework7 позволяет разработчикам создавать приложения, которые могут работать на разных платформах, используя один и тот же код. Это значительно упрощает и ускоряет процесс разработки, так как разработчики могут избежать необходимости создавать отдельный код для каждой платформы. Кроме того, использование веб-технологий позволяет разработчикам использовать свои навыки и знания в области веб-разработки для создания мультиплатформенных приложений.

В целом, использование Framework7 является одним из наиболее эффективных способов создания мультиплатформенных приложений, которые могут работать на разных платформах и устройствах. Благодаря его гибкости и мощным инструментам, разработчики могут создавать приложения с пользовательским интерфейсом, максимально приближенным к нативным приложениям для iOS и Android, используя открытые веб-стандарты.

Конфликт интересов

Не указан.

Рецензия

Все статьи проходят рецензирование. Но рецензент или автор статьи предпочли не публиковать рецензию к этой статье в открытом доступе. Рецензия может быть предоставлена компетентным органам по запросу.

Conflict of Interest

None declared.

Review

All articles are peer-reviewed. But the reviewer or the author of the article chose not to publish a review of this article in the public domain. The review can be provided to the competent authorities upon request.

Список литературы на английском языке / References in English

1. Kaur N. Comparative Analysis of Hybrid Mobile App Frameworks / N. Kaur, A. Singh // International Journal of Computer Applications. — 2017. — 168(2). — P. 1-5.

2. Madiajagan A. Framework7: A Cross-Platform Mobile Application Development Framework / A. Madiajagan, P.M. Lakshmi // 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI). — 2018. — P. 606-610.

3. Raj S. A Comparative Study of Hybrid Mobile App Development Frameworks / Raj S., Sharma R., Singh G. // Journal of Telecommunication, Electronic and Computer Engineering. — 2018. — 10(2-6). — P. 93-97.

4. Kumar A. Performance Evaluation of Hybrid Mobile App Frameworks / A. Kumar, R. Shastri // International Journal of Computer Science and Information Technology Research. — 2019. — 7(1). — P. 65-74.

5. Akter S. Hybrid Mobile Application Development: A Systematic Review / S. Akter, M.H.U. Rashid // 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS). — 2020. — P. 383-388.

6. Pal S.K. Comparative Analysis of Hybrid Mobile App Development Frameworks / S.K. Pal, V. Sharma // International Journal of Intelligent Enterprise. — 2021. — 8(2). — P. 138-153.

7. Das S. A Comparative Study of Hybrid Mobile App Development Frameworks / S. Das, D. Deb // Advanced Computing and Communication Technologies — 2021. — P. 397-409.

8. Griffith C. Mobile App Development with Ionic, Revised Edition: Cross-Platform Apps with Ionic, Angular, and Cordova / C. Griffith. — 2019.

9. Panhala M. Beginning Hybrid Mobile Application Development / M. Panhala. — 2018.

10. Akopkokhyants S. Mastering Framework 7: Build Multi-Platform Native Mobile Apps / S. Akopkokhyants, A. Ivanov. — 2017.