УДК 004

Разработка мобильного приложения с использованием Ionic 3

Вавилов Егор Дмитриевич
Приамурский государственный университет имени Шолом-Алейхема
Студент

Аннотация
В данной статье описан процесс создание мобильного приложение со списком дел. Для разработки используется мобильный SDK – Ionic. Конечным результатом является разбор создания приложения и запуск приложения.
Ключевые слова: Ionic, Приложение, Разработка

Mobile app development using Ionic 3

Vavilov Yegor Dmitrievich
Sholom-Aleichem Priamursky State University
student

Abstract
This article describes the process of creating a mobile to-do list app. Mobile SDK - Ionic is used for development. The end result is to parse the application creation and launch the application.
Keywords: Ionic, App, Development

Ionic - это мобильный SDK с открытым исходным кодом, который помогает создавать гибридные мобильные приложения. Он использует HTML, CSS и JavaScript. Ionic построен на основе Angular framework. Ionic использует библиотеку Cordova для доступа к встроенным аппаратным функциям, таким как доступ к камере, Bluetooth и другим.

Цель исследования – разработать мобильное приложение в котором можно добавлять и удалять поля с наименованием и запустить его.

Исследованиями в данной теме занимались следующие авторы. А.Н.Аблякимова, А.Н.Абдуллаев рассмотрели проблемы оптимизации кросс-платформенной разработки мобильных приложений[1]. Д.С.Сергеевич описал в своей работе основные аспекты по созданию мобильного приложения на основе Ionic[2]. Б.Н.Зыонг в статье рассмотрел способ установки приложений на устройство под управлением ОС Google Android[3]. Н.В.Панченков привел сравнительный обзор параметров разработки web-приложений и мобильных приложений [4].

Поскольку фреймворк Ionic построен на основе «Node», то необходимо установить «Node.js» для разработки приложений Ionic. После установки узла можно проверить ее версию с помощью команды «node –v» . Также для

проверки установленного npm - менеджера пакетов можно использовать команду «npm –v».

Далее устанавливаем Ionic CLI с помощью команды «npm install –g ionic». И так же проверяем с помощью команды «ionic -v».

Теперь чтобы создать новое приложение, используем команду «ionic start testApp»

Эта команда сгенерирует проект в текущем каталоге. Так же можно выбрать шаблон для начала из доступных. Некоторые из официальных доступных шаблонов:

- blank: простой шаблон со страницей.
- tabs: шаблон с 3 вкладками
- sidemenu: шаблом с боковым меню.
- super: шаблон с различными готовыми дизайнами.
- tutorial: шаблон обучающего проекта.

Для вывода списка всех доступных шаблонов используется команда «ionic start -list».

Для того, чтобы проверить созданное приложение, необходимо перейти в каталог приложения и выполнить команду «ionic serve». Эта команда откроет данное приложение в web-браузере (рис.1).

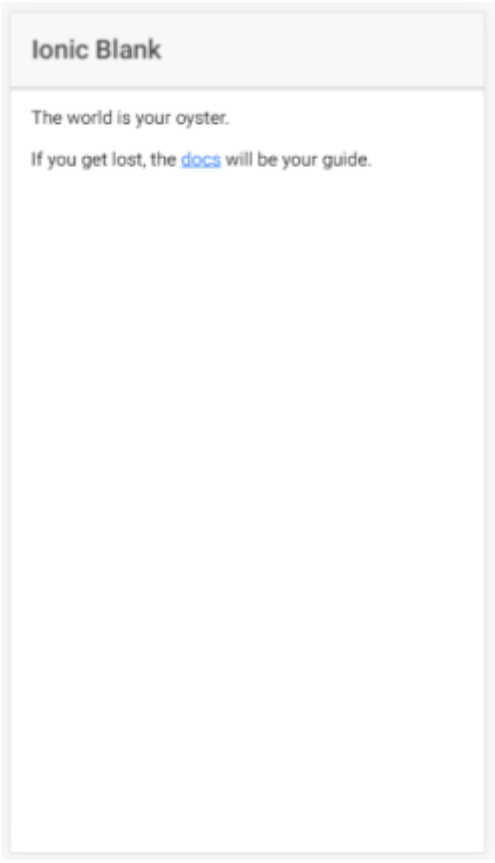

Рисунок 1 – Открытие приложения

Чтобы просмотреть приложение в эмуляторе или на устройствах, сначала нужно установить соответствующие платформы. Для этого

необходимо добавить целевые платформы, перейдем в созданный каталог проекта «ionic» и используем команду «ionic cordova platform add [platformname]» и вместо «platformname» указываем ее имя (android/ ios/ windows).

Для просмотра приложения в эмуляторе используем команду «ionic cordova emulate [platformname]», а для запуска на реальных устройствах «ionic cordova run [platformname]».

При создании использовались различные версии программ (рис.2).

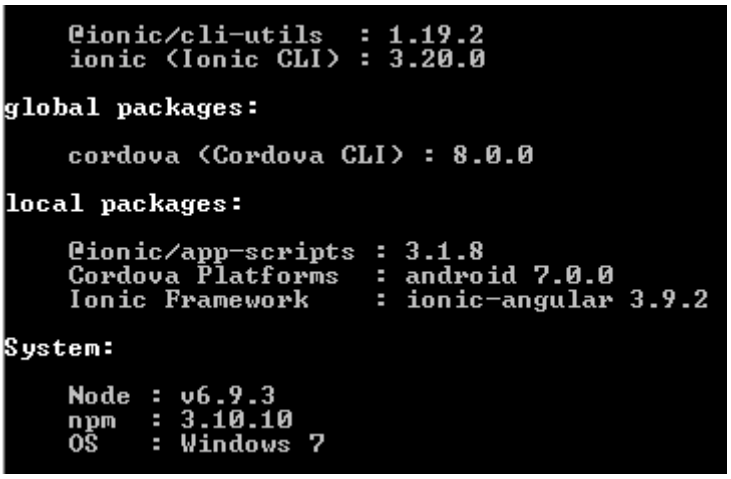

Рисунок 2 – Версии программ

Структура проекта будет выглядеть следующим образом (рис.3).

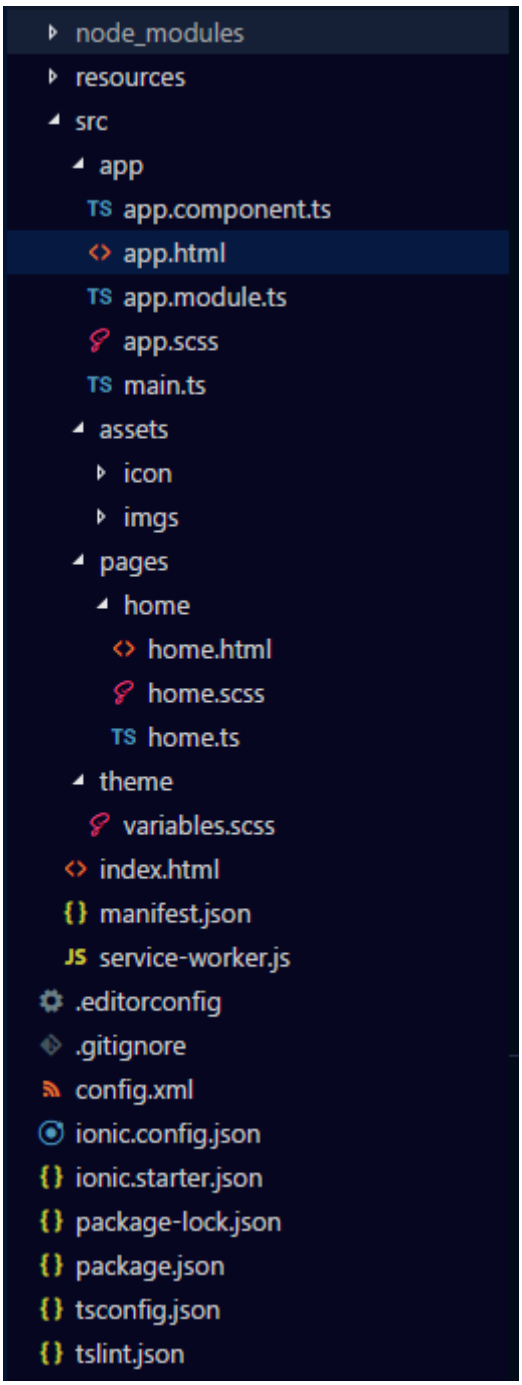

Рисунок 3 – Структура.

1. app.module.ts –В этом файле будут находиться и связываться множетво ресурсов, такие как страницы, модули, постановщики.
2. app.components.ts – Это страница запуска приложения, его код начинает обрабатываться сразу при запуске проекта.
3. app.html - шаблон приложения, к которому будут подключаться другие страницы пользовательского интерфейса.
4. app.scss - страница, содержащая все переменные и стили Sass, которые будут использоваться глобально в приложении.

Для начала займемся пользовательским интерфейсом. Напишем в файле «home.html» этот код (рис.4).

```
<ion-header>
    <ion-navbar>
        <ion-title>Ionic Blank</ion-title>
    </ion-navbar>
</ion-header>
<ion-content padding>.
    <p>Текст наполнения.
    </p>
</ion-content>
```

Рисунок 4 – домашняя страница

Добавим поле ввода и кнопку (рис.5).

```
<ion-content padding>
    <ion-item>
        <ion-input type="text" placeholder="Enter task" [(ngModel)]="taskName"/>
        <div class="item-note" item-end>
            <button ion-button>Введите значение</button>
        </div>
    </ion-item>
</ion-content>
```

Рисунок 5 – Поле ввода

Теперь, когда пользовательский интерфейс готов, перейдем к предоставлению ему функции. Запишем в файл home.ts код (рис.6).

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';

@Component({
    selector: 'page-home',
    templateUrl: 'home.html'
})

export class HomePage {
    constructor(public navCtrl: NavController) {  }
}
```

Рисунок 6 – home.ts

Здесь происходит импорт компонентов или внешних модулей, которые могут понадобиться на этой странице в самом верху. Следующие несколько строк описывают шаблон, которому принадлежат многие функции, которые можно написать, и которыми можно управлять. И, наконец, вся логика, которую можем кодировать.

Поскольку будут добавляться новые задачи каждый раз, нужно место для их хранения. Самый простой способ сделать это - инициализировать массив.

Назовем список «taskList», но поскольку нужен доступ к списку из более чем одного метода кода, нужно инициализировать его вне конструктора «taskList = [];». Теперь, чтобы написать код для обработки нажатия кнопки «Добавить задачу», назовем его «addTask». Все, что нужно сделать, это захватить текст во входных данных и поместить его в массив. Также необходимо убедиться, что в список не добавляется пустая строка, поэтому обернем приведенный выше оператор в условие «if», проверяя, действительно ли существует «taskName». Последний код home.ts (рис.7).

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';

@Component({
    selector: 'page-home',
    templateUrl: 'home.html'
})

export class HomePage {
    taskList = [];

    constructor(public navCtrl: NavController) {}

    addTask() {
        if (this.taskName.length > 0) {
            let task = this.taskName;
            this.taskList.push(task);
            this.taskName = "";
        }
    }
}
```

Рисунок 7 - home.ts

На этом все, работа над проектом закончена, осталось запустить и проверить его работоспособность (рис.8).

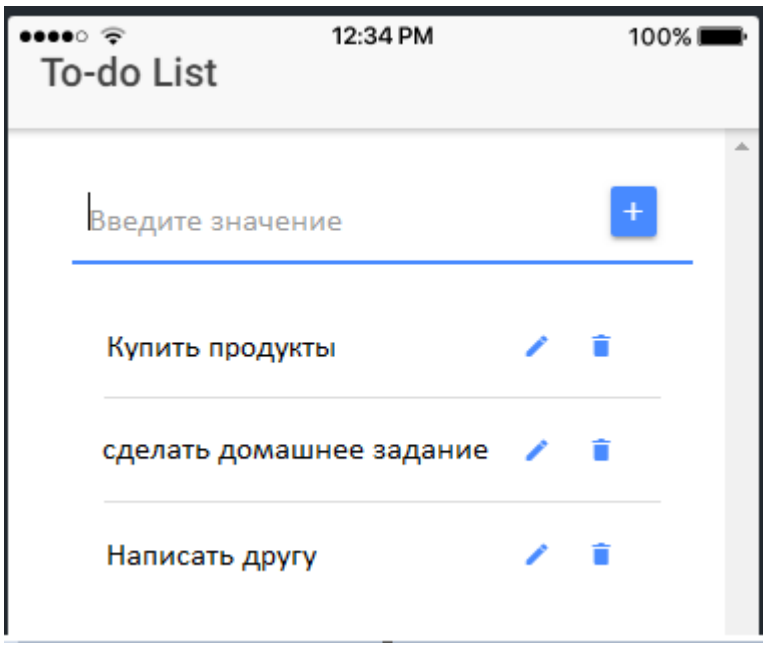

Рисунок 8 – Готовое приложение

В данной статье было разработано приложение со стартовым шаблоном и доработано до рабочего приложения со списком дел, где имеется возможность добавлять и удалять значения. Так же оно было протестировано с помощью браузера.

Библиографический список

1. Аблякимова А.Н., Абдуллаев А.Н. Ionic framework в разработке гибридных мобильных приложений // Информационно-компьютерные технологии в экономике, образовании и социальной сфере. 2018. №4(22). С. 62-68.
2. Сергеевич Д.С. Создание ionic чата для мобильных устройств при помощи react.js // Научные достижения и открытия современной молодёжи. 2019. №3. С. 58-62.
3. Зыонг Б.Н. Разработка установки мобильного приложения на android-устройстве в системе управления мобильными приложениями // Юность и знания - гарантия успеха - 2015. 2015. №1. С. 31-34.
4. Панченков Н.В. Особенности разработки веб-приложений и мобильных приложений // Тенденции развития науки и образования. 2019. №57-2. С. 24-26.