**ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ**

**Ордена Трудового Красного Знамени государственное образовательное учреждение высшего образования**

**Московский технический университет связи и информатики**

Кафедра Математической кибернетики и информационных технологий

РЕФЕРАТ

по дисциплине «Кроссплатформенные технологии программирования»

на тему: «Создание кроссплатформенных приложений с помощью React»

Выполнили:

При поддержке ДеДа ХаСаНа

Тешаев Парвиз Рустамович

Казан Максим Александрович

Разин Алексей Юрьевич

Студенты группы БСТ1703

Факультета ОТФ-2

Преподаватель: Полянцева Ксения Андреевна

**Оглавление**

[Введение 3](#_Toc5912804)

[Что такое React? 3](#_Toc5912805)

[Действительно нативный 4](#_Toc5912806)

[Преимущества для разработчика 5](#_Toc5912807)

[Обработка нескольких платформ 6](#_Toc5912808)

[Работа с React Native 7](#_Toc5912809)

[Начало 7](#_Toc5912810)

[Обычные компоненты React 7](#_Toc5912811)

[Таблицы стилей 9](#_Toc5912812)

[Настройка для мобильной разработки 9](#_Toc5912813)

[Погружаемся в нативный код 10](#_Toc5912814)

[Развёртывание приложения 12](#_Toc5912815)

[Заключения и рекомендации 13](#_Toc5912816)

# Введение

React Native — это JS-фреймворк для создания нативно отображаемых iOS- и Android-приложений. В его основе лежит разработанная в Facebook JS-библиотека React, предназначенная для создания пользовательских интерфейсов. Но вместо браузеров она ориентирована на мобильные платформы. Иными словами, если вы веб-разработчик, то можете использовать React Native для написания чистых, быстрых мобильных приложений, не покидая комфорта привычного фреймворка и единой кодовой базы JavaScript.

Конечно, мы и раньше слышали обещания об универсальной разработке приложений, с помощью фреймворков наподобие Cordova или Titanium. А что насчёт React Native? В этой статье мы рассмотрим данный фреймворк и особенности его работы и поговорим о том, насколько удобно использовать React Native для написания iOS- и Android-приложений.

## Что такое React?

React — это JS-библиотека для создания пользовательских интерфейсов, обычно для веб-приложений. Она разработана в Facebook и распространяется под лицензией open source с 2013 года. React широко распространена, и в отличие от более крупных MVC-фреймворков решает относительно узкую задачу: *рендеринг*интерфейса.

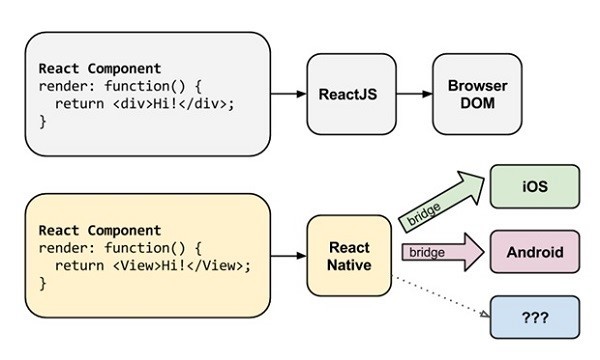
Популярность React имеет ряд причин. Она компактна и имеет высокую производительность, особенно при работе с быстроменяющимися данными. Благодаря компонентной структуре, React поощряет к написанию модульного, переиспользуемого кода.

React Native — это та же React, но для мобильных платформ. У неё есть ряд отличий: вместо тега divиспользуется компонент View, а вместо тэга img — Image. Процесс разработки остался тем же. Вам может пригодиться знание Objective-C или Java. Кроме того, в мобильной разработке есть свои подвохи (протестировал ли я на разных устройствах? Достаточно ли крупные объекты, чтобы на них комфортно нажимать?). Тем не менее, если вы работали с React, то React Native покажется вам практически такой же, столь же комфортной.

## Действительно нативный

Первое, что удивляет людей в React Native — он «действительно» нативный. Другие решения JavaScript-для-мобильных-платформ просто оборачивают ваш JS-код в хвалёное веб-представление. Они могут перереализовать какое-нибудь нативное поведение интерфейса, например, анимацию, но всё же это остаётся веб-приложение.

В React компонент описывает собственное отображение, а затем библиотека обрабатывает для вас рендеринг. Эти две функции разделены ясным уровнем абстракции. Если нужно отрисовать компоненты для веба, то React использует стандартные HTML-тэги. Благодаря тому же уровню абстракции — «мосту» — для рендеринга в iOS и Android React Native вызывает соответствующие API. В iOS компоненты отрисовываются в настоящие UI-виды, а в Android — в нативные виды.

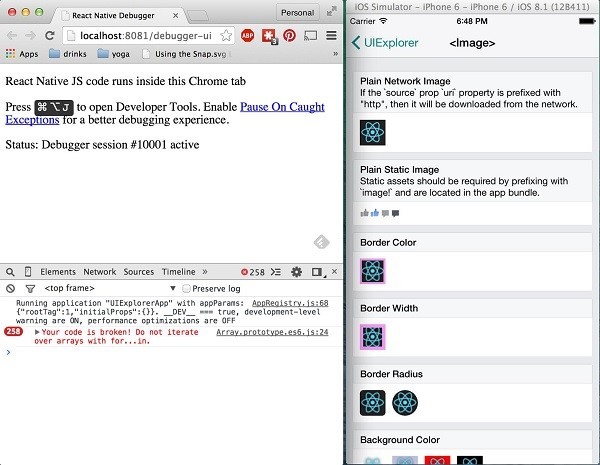


Вы будете писать ужасно выглядящий код, очень похожий на стандартный JavaScript, CSS и HTML. Вместо компилирования в нативный код, React Native берёт ваше приложение и запускает его с помощью JS-движка хост-платформы, без блокирования основного UI-потока. Вы получаете преимущества нативных производительности, анимации и поведения без необходимости писать на Objective-C или Java. Другие методы разработки кроссплатформенных приложений, вроде Cordova или Titanium, никогда не достигнут такого уровня нативной производительности или отображения.

# Преимущества для разработчика

По сравнению со стандартной разработкой под iOS и Android, React Native имеет гораздо больше преимуществ. Поскольку ваше приложение по большей части состоит из JavaScript, вы можете пользоваться многочисленными достоинствами веб-разработки. Например, чтобы увидеть внесённые в код изменения, можно мгновенно «обновить» приложение вместо длительного ожидания завершения традиционного ребилда. Похоже на дар свыше.

Кроме того, React Native предоставляет «умную» систему сообщений об ошибках и стандартные инструменты отладки JavaScript, что сильно облегчает процесс мобильной разработки.



## Обработка нескольких платформ

React Native изящно обрабатывает разные платформы. Подавляющее большинство API во фреймворке — кроссплатформенные, так что достаточно просто написать компонент React Native, и он будет без проблем работать в iOS и Android. В Facebook заявляют, что в их приложении Ad Manager 87% кода используется на обеих платформах.

Если вам нужно писать зависящий от платформы код — в связи с разными правилами взаимодействия в iOS и Android, либо из-за преимуществ платформозависимого API — то с этим не будет трудностей. React Native позволяет назначать платформозависимые версии каждого компонента, которые вы можете потом интегрировать в своё приложение.

# Работа с React Native

Нет ничего сложного в том, чтобы писать настоящее нативное приложение под iOS и Android, используя единую кодовую базу JavaScript. Как строится работа с React Native?

## Начало

В первую очередь нужно установить обычные зависимости для iOS- и Android-разработки, а также сам React Native. Хорошую инструкцию можно найти на сайте фреймворка. React Native прост в настройке. Если у вас уже установлена обновлённая версия Node, то можете установить фреймворк с помощью команды npm install -g react-native-cli.

После установки зависимостей запустите react-native init ProjectName. Будет автоматически сгенерирован шаблон нового проекта.

Есть одна уловка: при разработке с помощью React Native вам понадобится OS X. Для создания iOS-приложений Apple заставляет использовать Mac, это неизбежное ограничение для большинства разработчиков. Если же вы планируете писать только Android-приложения, то в React Native реализована экспериментальная поддержка разработки на Windows и Linux.

## Обычные компоненты React

После настройки окружения приступаем к написанию приложений. Как уже упоминалось, React Native — это просто React с несколькими основными отличиями. С точки зрения браузера, компоненты React Native выглядят очень похожими на компоненты React, но основные строительные блоки уже другие. Вместо таких тэгов, как div, img и p в React Native используются базовые компоненты вроде Text и View. В следующем примере используются компоненты ScrollView, TouchableHighlight и Text, применяемые для работы с видами, характерными для Android и iOS. C их помощью можно легко создать scrolling view, правильно обрабатывающий касания:

// iOS & Android

var React = require('react-native');

var { ScrollView, TouchableHighlight, Text } = React;

var TouchDemo = React.createClass({

render: function() {

return (

<ScrollView>

<TouchableHighlight onPress={() => console.log('pressed')}>

<Text>Proper Touch Handling</Text>

</TouchableHighlight>

</ScrollView>

);

},

});

Если вам ещё не приходилось иметь дела с JSX — мешаниной из HTML-образного синтаксиса и JavaScript — то этот код может показаться вам странным. И React, и React Native заставляют использовать JSX. Код рендеринга приложения соседствует с JavaScript, который управляет его поведением. Это часто вызывает недоумение у новичков, но я очень рекомендую не отвергать с ходу, а попробовать.

Поскольку компоненты React Native очень похожи на обычные компоненты React, вам будет очень просто перейти на этот фреймворк.

## Таблицы стилей

Для упрощения рендеринга и повышения его эффективности, а также для сохранения поддерживаемости кода, в React Native реализована ограниченная поддержка CSS. Вам не нужно изучать специфические способы разработки видов для каждой платформы, вместо этого достаточно освоить использование стилей React Native.

Основное различие заключается в том, что вам можно не волноваться о конкретных правилах. Здесь жёстко ограничено наследование стилей и используется inline-синтаксис.

Пример создания таблицы стилей в React Native:

var styles = StyleSheet.create({

container: {

flex: 1,

marginTop: 30

}

});

Применение стиля с помощью inline-синтаксиса:

<View style={styles.container}>

...

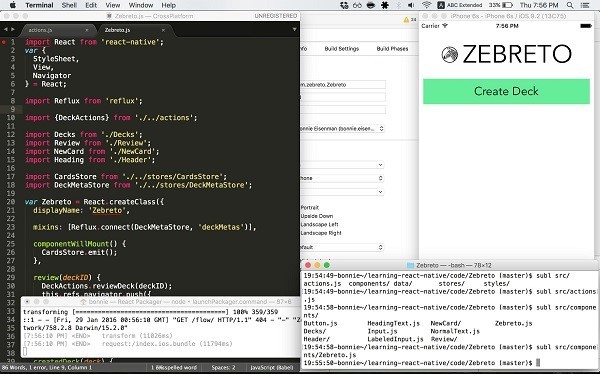
</View>

Синтаксис прост в чтении, но если раньше вы занимались веб-разработкой, то он покажется вам весьма настораживающим (и тому есть причина!). Из презентации Кристофера Шиду вы можете больше узнать о проблемах CSS и их решении в React.

## Настройка для мобильной разработки

Это более сложная часть работы с React Native. При работе с фреймворком вам понадобятся все обычные инструменты мобильного разработчика, а также инструменты для работы с JavaScript: текстовый редактор и средства отладки.

Для iOS у вас должен быть запущен Xcode и мобильный эмулятор, а для Android — Android Studio и ряд инструментов, запускаемых из командной строки. Наконец, вам понадобится и упаковщик React Native. Выбор текстового редактора для вашего JavaScript-кода остаётся на ваше усмотрение.



В общем, под рукой у вас должно быть много инструментов. Иногда кажется, что слишком много, и обилие открытых окон начинает раздражать. С другой стороны, React Native хотя бы не скрывает от вас ни одного стандартного процесса мобильной разработки.

## Погружаемся в нативный код

React Native обеспечивает JS-интерфейс для существующих API платформы. То есть вы можете писать практически такой же код, как в случае с React, а об остальном позаботится «мост» React Native. Но что если его окажется недостаточно?

В любом новом фреймворке неизбежны вызовы API, которые не поддерживаются этим фреймворком. В случае с React Native вы можете написать «нативный модуль» для взаимодействия между хост-платформой и вашим JS-кодом. Вот пример модуля Objective-C “Hello, World”:

// Objective-C

#import "RCTBridgeModule.h"

@interface MyCustomModule : NSObject <RCTBridgeModule>

@end

@implementation MyCustomModule

RCT\_EXPORT\_MODULE();

// Available as NativeModules.MyCustomModule.processString

RCT\_EXPORT\_METHOD(processString:(NSString \*)input callback:(RCTResponseSenderBlock)callback)

{

callback(@[[input stringByReplacingOccurrencesOfString:@"Goodbye" withString:@"Hello"]]);

}

@end

Чтобы использовать нативный модуль из JavaScript, его нужно запросить как любую библиотеку:

// JavaScript

var React = require('react-native');

var { NativeModules, Text } = React;

var Message = React.createClass({

getInitialState() {

return { text: 'Goodbye World.' };

},

componentDidMount() {

NativeModules.MyCustomModule.processString(this.state.text, (text) => {

this.setState({text});

});

},

render: function() {

return (

<Text>{this.state.text}</Text>

);

}

});

Так можно сделать, если:

* нужный вам API ещё не поддерживается,
* вы хотите интегрировать свои React Native-компоненты с существующим кодом на Objective-C или Java,
* вам нужно написать какую-то высокопроизводительную функцию для обработки интенсивных графических вычислений.

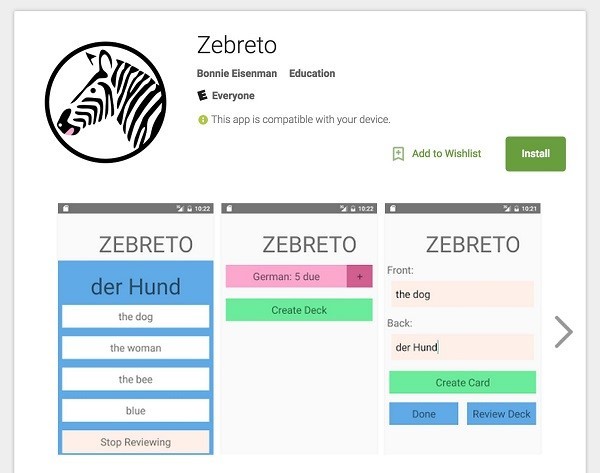
К счастью, React Native достаточно гибок и позволяет писать так называемые «нативные модули». Даже если вам не приходилось раньше работать с Objective-C или Java, написание кода «моста» будет отличным упражнением по освоению нативной мобильной разработки.

## Развёртывание приложения

В React Native развёртывание выполняется почти так же, как и в случае с обычными приложениями. То есть проходит непросто, поскольку процесс выпуска мобильных приложений известен своими заморочками.

Для создания готового к развёртыванию пакета вам нужно использовать пакетный (bundled) JavaScript вместо версии, поддерживающей live reloading разработку. Для этого в iOS нужно внести одну строку в файл AppDelegate.m и выполнить react-native bundle --minify. А в Android нужно просто выполнить ./gradlew assembleRelease. В остальном процесс пакетирования на обеих платформах такой же, как и в случае с обычными мобильными приложениями. В результате вы получите пакет, который готов к размещению в соответствующем магазине приложений.

Обычно утверждение размещения React Native-приложений в магазинах занимает столько же времени, сколько и для «традиционных» приложений. Например, приложение Zebretto утверждалось в AppStore в течение двух недель, а в Play Store — меньше одного дня.



Apple позволяет приложениям обновляться самостоятельно, если изменения касаются только JavaScript. Это избавляет от ряда проблем при развёртывании. Microsoft недавно выпустила SDK CodePush, позволяющий разработчикам, использующим React Native, мгновенно выпускать обновления. Очень соблазнительная функция, наверняка скоро всё больше приложений начнут её использовать.

# Заключения и рекомендации

Если у вас есть опыт веб-разработки, то вам должен понравиться React Native. Этот фреймворк превращает любого веб-разработчика в потенциального разработчика мобильных приложений, позволяя значительно улучшить процесс их создания.

У React Native есть свои недостатки. Это относительно новый проект, и ему свойственны проблемы всех молодых библиотек: отсутствуют некоторые функции, ещё не выработаны оптимальные методики использования. От релиза к релизу внедряются серьёзные изменения, хотя их немного и они носят ограниченный характер.

Тем не менее, React Native уже достаточно зрелый проект, чьи достоинства весомее недостатков. С помощью этого фреймворка можно использовать единую кодовую базу для создания приложений под iOS и Android, не жертвуя ни качеством, ни производительностью. Даже если вы не имеете опыта в JavaScript, вряд ли будете оспаривать преимущества более быстрого цикла разработки и почти стопроцентного переиспользования кода. А поскольку React Native позволяет при необходимости переходить на «нормальную» разработку, то вам не мешают свойственные фреймворку ограничения. В общем, React Native обеспечивает высококачественную кроссплатформенную мобильную разработку, и к этому инструменту стоит серьёзно присмотреться.

# Список литературы

1. Бонни Эйзенман. Написание кроссплатформенных приложений с помощью React Native [Электронный ресурс]: статья. – / Бонни Эйзенман – Электрон. текстовые дан. – [б.и.], 2016 – Режим доступа: <https://www.infoq.com/articles/react-native-introduction>