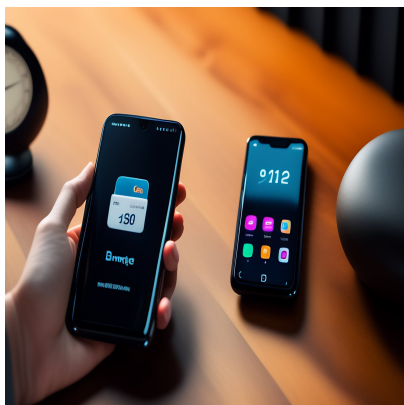




14 языков программирования для разработки мобильных приложений

Описание

Много лет назад существовали только варианты создания мобильного приложения – один код для приложений iOS и один код для Android. Но сегодня у разработчиков гораздо больше возможностей для создания приложений. Я создал это руководство, чтобы помочь вам понять наиболее популярные языки программирования и фреймворки для разработки мобильных приложений. Вы можете использовать эту информацию, чтобы решить, какой язык изучать и во что инвестировать для своего мобильного приложения.

В рамках данного ресурса я не буду рассматривать типы конструкторов мобильных приложений. Технически, вам не нужно изучать языки программирования для этого типа платформы разработки мобильных приложений. Ознакомьтесь с нашим руководством по пяти способам создания приложений для получения дополнительной информации об этих альтернативных методах.

Типы мобильных приложений

Прежде чем мы продолжим, я просто хочу быстро рассказать о различных типах разработки приложений с точки зрения кодирования. Вообще говоря, разработчики мобильных приложений могут создавать приложения в одной из этих трех категорий:

Нативные приложения

Разработка нативных приложений ведется на языке, который поддерживается конкретной операционной системой мобильного устройства. (Пример: нативное приложение для iOS против нативного приложения для Android). Это используется, если вы создаете приложение специально для Apple App Store или Google Play Store. Это идеальный вариант для приложений с высоким уровнем кастомизации, которые должны использовать “родные” компоненты каждого устройства. Он отлично подходит для игровых приложений, приложений VR и приложений с объемной графикой. Однако при нативной разработке один код не будет работать на обеих платформах.

Гибридные приложения

Гибридные приложения предназначены для кроссплатформенной разработки. Они написаны на одном языке, который может работать на нескольких платформах – iOS и Android. Это ускоряет сроки разработки, так как вам придется писать все только один раз, а не дважды (по одному разу для каждой ОС). По сравнению с нативным, вы потеряете немного гибкости в плане того, что вы можете делать с гибридными приложениями. Но это устраивает подавляющее большинство разработчиков.

Самая распространенная тема, которую можно услышать при обсуждении разработки мобильных приложений, – это разница между нативными и гибридными приложениями. Это стало большой дискуссией для разработчиков уже довольно давно. Как я уже кратко упоминал выше, нативные приложения создаются для конкретной операционной системы. Поэтому если вы хотите разработать приложение для iOS и Android, вам придется создавать его отдельно для iOS и отдельно для Android. У этого метода, как и у других, есть свои плюсы и минусы. Мы подробнее рассмотрим преимущества и недостатки этих методов разработки приложений по мере продвижения по этому руководству.

PWA-приложения

PWA (progressive web app) – это легкое приложение, которое запускается в URL браузера устройства. Оно выглядит и ощущается как мобильное приложение, но не поставляется на устройство. Разработчики, имеющие опыт веб-разработки, могут

легко создавать PWA. Вы уже должны быть знакомы с языками кодирования, используемыми в процессе разработки. Таким образом, обучение будет не таким сложным. С другой стороны, эти приложения будут немного более ограниченными в плане возможности использования компонентов родного устройства.

Лучшие языки программирования для разработки приложений

Давайте подробнее рассмотрим 14 лучших языков программирования для мобильных приложений. Каждый вариант имеет свои преимущества и недостатки в зависимости от уровня вашей квалификации и предполагаемого сценария использования. Независимо от вашего оборудования и программного обеспечения, вы можете найти кодеров и языки, которые соответствуют вашим потребностям.

Языки программирования iOS

Платформа iOS была создана компанией Apple. Если вы разработаете приложение для iOS, оно будет работать в экосистеме Apple, например, на iPhone и iPad. Приложения, созданные с использованием языка программирования iOS, могут быть доступны в Apple App Store для загрузки пользователями. Для того чтобы создать приложение для iOS, вам необходимо иметь учетную запись разработчика Apple. Вам также понадобится среда разработки Xcode, установленная на компьютере Mac (на компьютере под управлением Windows вы не сможете создавать и отлаживать приложения должным образом).

Xcode поставляется со всем необходимым для создания приложений для всех устройств Apple. В этом наборе инструментов разработки есть редактор кода, симуляторы, отладчик и SDK. Существует два родных языка программирования для разработки iOS – Objective-C и Swift.



Ниже мы рассмотрим каждый из этих языков программирования Apple более подробно.

Objective-C

Objective-C был первым языком программирования компании Apple для поддержки мобильных приложений на своей платформе. Это ОО (объектно-ориентированный язык программирования), который использует синтаксис языка C и объектно-ориентированные аспекты языка SmallTalk. Язык не очень удобен для разработчиков. Одним из недостатков является то, что синтаксис кажется неуклюжим, а квадратные скобки могут быть сложными для отладки.

Swift

Swift был представлен в 2014 году как язык программирования Apple. В следующем году он стал доступен для разработки в Xcode. Этот язык быстро стал предпочтительным для разработчиков при создании приложений для iOS. Если вы

только начинаете разрабатывать приложения для iOS, это один из лучших языков программирования, который стоит рассмотреть. Использование Objective-C снизилось после появления Swift. Для любых современных приложений, создаваемых на платформе Apple, Swift очень рекомендуется. По сравнению с Objective-C и другими языками программирования, Swift проще и компактнее. У любого разработчика Apple, который уже умеет создавать приложения на Objective-C, не должно возникнуть проблем с переходом на Swift.

Языки программирования Android

Android – это платформа для разработки программного обеспечения с открытым исходным кодом, которой управляет компания Google. Хотя у Google есть свои собственные варианты мобильных устройств для телефонов и планшетов, другие производители, такие как Samsung, Huawei, Microsoft и другие, также выпускают телефоны и планшеты, работающие на базе ОС Android.

Чтобы создать приложение для Android, необходимо получить набор инструментов для разработки Android, в котором есть отладчики, эмуляторы и необходимый SDK. Лучшей интегрированной средой разработки (IDE) для разработки приложений Android является Android Studio. Это упрощает задачу, если вы разрабатываете мобильные приложения для Android. Существуют и другие варианты, включая кроссплатформенные инструменты разработки приложений (для создания приложений для Android и iOS, а также мобильных приложений для других операционных систем), но Android Studio определенно является самой популярной для разработки приложений для Android. IDE для Android, как правило, можно запускать на любой ОС, включая Windows, Mac и Linux.



Давайте подробнее рассмотрим два языка программирования для Android – Java и Kotlin.

Java

С момента официального запуска Android в 2008 году язык Java является языком разработки по умолчанию для написания приложений для Android. Этот объектно-ориентированный язык был создан еще в 1995 году. Хотя Java имеет свою долю недостатков, он по-прежнему является самым популярным языком для разработки Android, поскольку работает на виртуальной машине. Как объектно-ориентированный вариант мобильной разработки, Java обычно используется для разработки приложений для Android. Большинство других языков Android считаются версией Java или разновидностью Java.

Kotlin

Google объявил, что начнет поддерживать язык программирования Kotlin в 2017 году. Это язык, альтернативный традиционному Java для разработки Android, и он работает на виртуальной машине Java. Даже будучи новым языком, он очень популярен. Kotlin и Java являются совместимыми, то есть они могут использовать одну и ту же информацию. Все ваши библиотеки Java могут быть доступны с помощью Kotlin. С точки зрения исполнения, язык Kotlin соответствует Java Bytecode. В целом, он считается более аккуратной и чистой версией Java.

Языки программирования для нативной разработки

Как я уже говорил, все языки программирования для нативных мобильных приложений имеют свои плюсы и минусы. Используете ли вы Objective-C или Swift для Apple или Java или другой вариант Java (например, Kotlin) для Android, вот преимущества и недостатки.

Плюсы нативного программирования:

- Наибольший контроль над устройством
- Низкоуровневое кодирование для передовых технологий, которые добавляются на устройство
- Самый быстрый доступ к новейшим и лучшим функциям через ваш язык
- Самая быстрая в исполнении итоговая линия

Минусы нативного программирования:

- Самая медленная разработка
- Самый дорогостоящий метод разработки
- Требуются высококвалифицированные и специализированные разработчики мобильных приложений для iOS и Android
- Высокий барьер для входа

Хотя родные языки программирования дают вам наибольший контроль над вашим приложением, их трудно изучить, а разработка занимает много времени. Если вы не создаете узкоспециализированное приложение, вам, скорее всего, не придется использовать родной язык программирования.

Гибридные языки программирования

Гибридные приложения разрабатываются один раз, но пишутся на языке программирования, который работает на нескольких платформах.



Чаще всего одна разработка работает как для iOS, так и для Android. Хотя некоторые гибридные языки расширяют свою функциональность на другие платформы, например, прогрессивные веб-приложения (PWA) или мобильные веб-приложения. Это полезно для тех, кто переходит на более дружелюбную веб-среду. Когда вы создаете гибридное приложение или хотите разрабатывать кроссплатформенные приложения, вы, как правило, имеете дело с каким-либо языком, фреймворком или набором инструментов на базе JavaScript. Гибридные приложения работают на различных мобильных устройствах. Давайте подробнее рассмотрим некоторые из этих вариантов ниже.

C#

Разработанный компанией Microsoft, C# (произносится как “Си шарп”) является еще одним объектно-ориентированным языком программирования. Это популярный язык программирования для разработки игр и сценариев командной строки для операционных систем Android. Другие альтернативные типы low-code, такие как OutSystems и Kony, имеют SDK, который можно использовать с разными языками, а не только с одним. Существуют и другие языки для разработки мобильных приложений, которые используют язык системного программирования с синтаксисом, похожим на C#. Используя IDE для гибридной разработки, код C# кросс-компилируется для запуска на устройствах iOS и Android.

Xamarin

Со временем Microsoft приобрела фреймворк Xamarin, который позволяет разработчикам приложений программировать на C# с использованием других фреймворков. Технически, Xamarin не является языком. Это платформа разработки с открытым исходным кодом для приложений iOS, Android и Windows. Это платформа .NET, использующая C# в качестве основного языка.

React Native

Reactive Native использует JavaScript для связи с предварительно созданной функциональностью, которая является родной для их фреймворка. Это позволяет вам манипулировать пользовательским интерфейсом, собирать данные и извлекать их, чтобы представить пользователю. По сути, это означает, что вы в значительной степени полагаетесь на JavaScript для манипулирования родными компонентами. Программирование мобильного приложения таким способом имеет свои плюсы и минусы.

Appcelerator

Подобно Reactive Native, Appcelerator также использует JS для связи функций с нативным фреймворком. Appcelerator Titanium позволяет создавать нативные приложения для iOS, Android и Windows.

Плюсы Appcelerator:

- Использование Java для манипулирования чем-то родным
- Доступ к нативной функциональности непосредственно из JavaScript

Минусы Appcelerator:

- Не использует такие вещи, как HTML5 или CSS – технологии, которые обычно используются всеми, кто разрабатывает на JavaScript, чтобы свободно манипулировать собственным пользовательским интерфейсом.

Барьер входа для изучения этого метода программирования находится примерно на среднем уровне.

Cordova/PhoneGap

Фреймворки типа Cordova/PhoneGap и Ionic на самом деле просто построены поверх языка программирования Apache Cordova. Гибридная разработка мобильных приложений с использованием этого языка программирования строится путем переноса веб-опыта в нативный опыт.

Что это значит?

Этот метод позволяет создавать приложения так же, как и веб-сайты. Так что если вы веб-разработчик, вы будете чувствовать себя здесь как дома. Здесь используются Javascript, HTML и CSS. Эта веб-среда переносится на iOS и Android. Готовое программное обеспечение будет работать на нескольких типах мобильных устройств.

Плюсы:

- Низкий барьер для входа
- Любой человек с опытом веб-разработки может легко начать программировать таким образом.
- Кривая обучения проста

Минусы:

- Небольшая потеря частоты кадров

Если вы разрабатываете мобильные игры, нативные приложения с машинным обучением или нативные приложения дополненной реальности (AR), это решение, вероятно, не лучший язык программирования для вас. И то, и другое требует более высокой частоты кадров.

Языки программирования PWA

Прогрессивные веб-приложения предлагают функции, подобные приложениям, на мобильном сайте. Это быстрая и надежная альтернатива традиционным мобильным веб-сайтам. В отличие от веб-сайта, веб-приложения могут работать в автономном режиме, а также получать доступ к встроенным возможностям устройства (таким как камера, GPS и т.д.). Вот некоторые языки программирования, которые можно использовать для PWA:

Ruby

Ruby – это язык программирования общего назначения, который можно использовать для широкого круга задач, включая PWA. Многие разработчики полагаются на Ruby при создании веб-приложений из-за его простоты. Этот язык существует с 1990 года и остается любимым в сообществе разработчиков и сегодня. Хотя отправлять код на Ruby легко, поиск ошибок и их отладка не всегда так просты. Так что имейте это в виду, если вы идете этим путем.

Python

Python – это еще один язык кодирования общего назначения. Это объектно-ориентированный язык, который дает разработчикам гибкость как для небольших, так и для масштабных развертываний. Помимо PWA, Python широко используется для анализа данных, визуализации данных, создания веб-сайтов, автоматизации задач и других типов программного обеспечения. Большинство разработчиков согласятся, что Python немного сложнее, чем Java. Он имеет более крутую кривую обучения по сравнению с другими языками в этой категории.

CSS

CSS (Cascading Style Sheets) – это язык, основанный на правилах. Он помогает описать представление кода, написанного на языках разметки, таких как HTML. Вы

не сможете создать PWA с помощью одного лишь CSS. Но вы, скорее всего, будете использовать его в качестве дополнения к JS или HTML-коду.

JavaScript

Если у вас есть опыт веб-разработки, использование JavaScript для создания PWA может оказаться для вас лучшим вариантом. Вы можете использовать JS поверх HTML и наряду с CSS для создания веб-приложения с нуля. По сравнению с другими вариантами PWA, этот вариант имеет более низкий барьер для входа для тех из вас, кто имеет некоторые базовые технические знания и опыт кодирования. Но он не такой гибкий, как Ruby или Python для PWA.

PHP

PHP – это общий язык сценариев, который был впервые представлен в 1994 году. Хотя у него есть несколько потенциальных вариантов использования, PHP также может быть использован для PWA. Однако это не обязательно лучший вариант. Вы можете использовать PHP в бэкенде для приложений, но вам все равно захочется использовать HTML, CSS и JS на фронт-энде. Использование PHP для простого создания веб-сайта не будет иметь такого же внешнего вида, ощущения и возможностей, как традиционный PWA. Короче говоря, один только PHP не обеспечит создание прогрессивного веб-приложения. Но большая часть вашей работы все равно может быть написана на PHP.

Выбор правильного языка программирования

Как видите, у каждого языка программирования есть свои плюсы и минусы. Нельзя однозначно сказать, что один из них лучше другого. Все зависит от типа приложения, бюджета, сроков и технических знаний. Нативная разработка, как правило, самая дорогая и самая сложная для изучения. Но она необходима для некоторых типов приложений, например, игровых.

Гибридные языки легче освоить пользователям, имеющим некоторые технические знания и опыт веб-разработки. С помощью этого метода вы сможете быстро вывести свое приложение на рынок, поскольку вам не придется использовать два варианта развертывания. Помимо выбора конкретного языка, вам нужно будет выбрать подход. У нас есть полное руководство, в котором описаны различия

между синхронным и асинхронным программированием, чтобы помочь вам сделать выбор между этими двумя популярными методами.

Когда использовать приложение для кодирования

Такие приложения для кодирования, как Grasshopper, Sololearn и Mimo, являются отличными вариантами для начинающих. Так что если вы только начинаете свой путь разработчика, эти инструменты будут вам полезны. Кроме того, вы можете использовать конструктор приложений, чтобы создать приложение, не написав ни строчки кода. Такие конструкторы приложений без кода, как BuildFire, позволяют любому человеку, независимо от технических знаний, создать приложение для iOS и Android одновременно.

Если вы не хотите стать разработчиком, использование создателя приложений – лучший вариант. Время, которое вам потребуется на изучение различных языков программирования, вероятно, не стоит вложений, если вам нужно создать только одно приложение для вашего бизнеса. Даже если вы научитесь кодировать, ваше первое приложение, скорее всего, не будет готово для реальных пользователей.

MBaaS (Mobile Backend as a Service)

Все гибридные фреймворки и родной язык программирования для создания мобильных приложений имеют две общие черты – все они должны быть созданы с нуля, и всем им не хватает одного важного компонента. Любой, кто разрабатывал мобильные приложения в прошлом, понимает, что само приложение – это лишь часть всей среды и общего решения. Вам также понадобится массивный мобильный бэкенд как сервис – более известный как MBaaS.

Для чего вам нужен MBaaS? Вот несколько примеров:

- Хранить ваши данные
- хранение профилей пользователей
- Составлять аналитику
- Отправка push-уведомлений

Список можно продолжать и продолжать. Все это – серверы в облаке, которые вам также необходимо разработать для поддержки вашего приложения. Если только вы не создаете простое приложение, например, калькулятор, вам, как правило, нужен

какой-то тип аутентификации пользователей, база данных, CMS и т.д.

BuildFire JS

Именно здесь в игру вступает BuildFire JS. Фреймворк BuildFire JS позволяет создавать приложения так же, как и на гибридной платформе Cordova. Вы можете использовать веб-технологии, такие как JavaScript, HTML и CSS. Но этот фреймворк не заставляет вас создавать все с нуля.



Такие вещи, как аутентификация и push-уведомления, строятся поверх существующей платформы. Эта платформа обладает всеми типичными функциями, которые необходимы большинству приложений, например, логины пользователей, функции сброса пароля, доступ к базам данных, доступ к платформам CMS и так далее. С BuildFire JS вам нужно создать только то, что уникально для вашего конкретного приложения. Серверы аналитики, базы данных, серверы push-

уведомлений, шлюзы API и многое другое – все это часть огромного MBaaS, предоставляемого BuildFire.

Все это находится в среде с открытым исходным кодом, что позволяет людям постоянно добавлять новые функции в платформу. Вы можете интегрировать эти функции в свое приложение, не беспокоясь о проблемах безопасности или лицензировании. После того, как все сказано и сделано, и вы разработали свое приложение с помощью BuildFire, есть внутренняя панель управления, которая позволяет вам управлять своим приложением по воздуху, не сталкиваясь с трудностями публикации и обновления. Поскольку ваше приложение создано на платформе с MBaaS, вам не придется беспокоиться о новых политиках, правилах, вопросах соответствия, функциях и многом другом на iOS и Android. BuildFire позаботится о том, чтобы ваше приложение всегда соответствовало требованиям.

Заключительные размышления: Лучшие языки программирования

Какой язык программирования лучше всего подходит для разработки мобильных приложений? Не существует правильного или неправильного ответа на вопрос, какой язык программирования вам следует изучать или в какой фреймворк вам следует инвестировать. Лучшие языки программирования для меня и моих мобильных приложений могут оказаться не самыми лучшими для вас и вашего сценария разработки приложений. Все варианты, перечисленные в этом руководстве, являются хорошими и правильными вариантами для рассмотрения. У каждого из них есть свои плюсы и минусы. Существуют даже дополнительные языки программирования для разработки мобильных приложений, например, Python для программирования на стороне сервера и другие.

Вам просто нужно выяснить, что лучше всего подходит для вас, вашего бизнеса и ваших целей. Какой тип приложения вы создаете? Что нужно приложению? Где вы хотите приложить больше всего усилий? Вы хотите разработать его один или несколько раз? На каком мобильном устройстве будет работать конечный пользователь? Вы создаете приложение для нескольких платформ? Как насчет контроля доступа к данным, статистически типизированного языка программирования или интерпретируемого языка программирования? Будете ли вы использовать какие-либо инструменты разработки с низким кодом/без кода? Вот некоторые из вопросов, которые вы должны задать себе, чтобы определить, где лучше всего использовать ваше время, усилия и ресурсы. В конце концов, просто

убедитесь, что вы сможете быстро выйти на рынок с лучшим из возможных приложений.

Дата Создания

13.04.2023