React Native vs Swift в 2023 году: Что лучше для разработки iOS-приложений?

Описание

Краткое резюме: Вы собираетесь создать свое первое приложение для iOS, но пытаетесь понять текущее мнение о React Native и Swift? Вы попали на нужную страницу, поскольку мы собираемся подробно сравнить React Native и Swift, чтобы вы получили исчерпывающие знания об этих двух технологиях и приняли лучшее решение. Итак, начнем:

Позвольте нам вынести свой вердикт:

- Если вы хотите создать нативное приложение для iOS, Swift – лучший вариант для вас. Если вы хотите заняться кроссплатформенной разработкой мобильных приложений, React Native – лучший вариант для вас.
- Но это вам скажет любой. Но вот что они вам не скажут, так это какой вариант идеально подходит для вашего проекта и почему.

Поэтому мы решили эту проблему за вас!

> React Native предлагает гибкость кроссплатформенной разработки, в то время как Swift обеспечивает производительность нативных приложений. Если вы наняли лучших разработчиков приложений React Native, вы, должно быть, ищете гибкость. А если вы наняли разработчиков Swift, то ваша главная цель – производительность.

Создание приложения для iOS – сложная задача, хотя для этого достаточно нанять

компанию по разработке приложений для iOS после обсуждения требований к проекту, стоимости и сроков, и получить его разработку. Для этого нужно проделать большую работу. Вам нужно изучить множество вещей, но выбор правильной технологии или платформы – это самый трудоемкий день. То же самое происходит и с компаниями, которые хотят создать приложение для iOS. Без сомнения, у них есть множество вариантов, но выбрать, какой из них подходит для проекта, – вот в чем суть игры. React Native vs. Swift поможет вам справиться с этой задачей.

В этой статье мы обсудим и сравним Swift и React Native и поймем, для какого проекта они лучше всего подходят. В первую очередь, в этой статье мы рассмотрели три основных момента:

1. Что такое React Native и Swift?
2. Сравнение React Native и Swift.
3. Помочь вам выбрать правильные технологии

Что такое Swift?

Swift – это самый известный язык программирования, разработанный компанией Apple для создания нативных приложений для iOS, macOS, watchOS и tvOS. Несомненно, предприятиям и предпринимателям приходится нести большие первоначальные затраты, но разработка нативных приложений для iOS является лучшим вариантом по сравнению с кроссплатформенной разработкой с точки зрения качества и производительности.

Согласно опросу, проведенному Stack Overflow, Swift занимает 9-е место среди самых популярных языков программирования среди разработчиков по всему миру благодаря своей гибкости, что делает эту платформу не только хорошим выбором для создания приложений для iOS и macOS. Она также используется для создания приложений на стороне сервера и моделей машинного обучения.

Что такое React Native?

React Native – это гибридный фреймворк для разработки мобильных приложений, разработанный компанией Facebook в 2015 году. По данным Statista, с долей рынка в 42% в области разработки мобильных приложений он занимает первое место как лучший кроссплатформенный фреймворк для разработки приложений. React Native

– это фреймворк с открытым исходным кодом и является лучшим вариантом для создания мобильного приложения для Android и iOS.

Лучшая часть этой технологии – доступность разработчиков, поскольку вы можете найти разработчиков React Native без особых усилий. React Native – это находка для предприятий и предпринимателей, которые хотят разрабатывать экономически эффективные мобильные приложения. Мобильное приложение, созданное с использованием React Native, является быстрым и масштабируемым, предоставляя разработчикам возможность один раз написать и развернуть приложение на платформах Android и iOS.

Зачем вообще сравнивать React Native и Swift?

И React Native, и Swift – отличные технологии, используемые для разработки мобильных приложений. Но они отличаются друг от друга во многих отношениях. И существенная разница между ними заключается в том, что первый является одним из лучших кроссплатформенных фреймворков, который позволяет разработчикам писать один раз и развертывать приложения как на платформах iOS, так и Android. В то же время, второй – это язык программирования, специально разработанный для разработки приложений для iOS.

Это сравнение только разряжает обстановку и сомнения и помогает предпринимателям (даже разработчикам) определить, какая цифровая инновация подходит для их работы. Например, если вы хотите за короткое время разработать мобильное приложение как для Android, так и для iOS, вы можете рассмотреть React Native. Разработчики приложений React Native пишут одну кодовую базу, которая может быть развернута на нескольких платформах, таких как iOS и Android. Однако если вы хотите создать мобильное приложение только для платформ iOS с высокой производительностью и сложным дизайном пользовательского интерфейса, Swift будет лучшим выбором.

Плюсы и минусы Swift и React Native

Обе технологии хороши своими элементами и лучше всего подходят для разработки мобильных приложений для iOS. Но, как мы уже говорили, они отличаются во многом; понимание их преимуществ и недостатков поможет вам разобраться в технологиях, и в конце этой статьи вы сможете сделать правильный выбор в пользу

лучшей из них.

Преимущества языка программирования Swift

Вот 7 лучших преимуществ языка программирования Swift;

- Современный и динамичный
- Безопасность и надежность
- Быстрее и мощнее
- Открытый исходный код
- Перспективный
- Интероперабельность
- Простота использования

Современный и динамичный

Это последняя инновация, но в ней сочетается опыт, накопленный за десятилетия работы Apple. Например, Swift поставляется с чистым синтаксисом. Это означает,что интерфейсы прикладного программирования (API) на этом языкепрограммирования легче читать и поддерживать. Давайте поймем это на примере:

```
struct Player
{
  var name: String
  var highScore: Int = 0
  var history: [Int] = []

  init(_ name: String)
  {
    self.name = name
  }
}
var player = Player("Tomas")
```

Безопасность и надежность

Swift разработан с учетом требований безопасности, в него включены такие функции, как необязательные представления и вывод типов, чтобы снизить риск распространенных ошибок программирования. Приведенный здесь пример показывает, насколько идеально он обеспечивает безопасность и надежность.

```
extension Collection where Element == Player {
  // Returns the highest score of all the players,
  // or `nil` if the collection is empty.
func highestScoringPlayer() -> Player?
  {
    return self.max(by:
    { $0.highScore < $1.highScore })
  }

}
```

Кроме того, необязательное связывание, цепочка и коалесцирование nil – это необязательные значения, которые обеспечивают безопасность. Вот пример того, как это происходит:

```
if let bestPlayer = players.highestScoringPlayer() {
  recordHolder = """
  The record holder is (bestPlayer.name),
  with a high score of (bestPlayer.highScore)!
  """
  }
  else
  {
```

```
    recordHolder = "No games have been played yet."
  } print(recordHolder)
  // The record holder is Erin, with a high score of 271!
  let highestScore = players.highestScoringPlayer()?.highScore ?? 0

// highestScore == 271
```

Быстрее и мощнее

Благодаря высокопроизводительной технологии компилятора LLVM, Swift работает так же быстро, как и более ранние технологии. В результате усилия разработчиков Swift помогают им преобразовать код в оптимизированный машинный код. Он создан для того, чтобы быть быстрым, и, следовательно, это делает его отличным выбором для создания высокопроизводительных приложений.

Открытый исходный код

Вам не нужно платить за использование этого языка программирования, так как он имеет открытый исходный код и доступен для скачивания. Просто наймите разработчиков приложений и получите готовые приложения. Помимо открытого исходного кода, Swift гарантирует разработчикам постоянную поддержку. Существует большое сообщество разработчиков Swift, которые постоянно работают с языком и предоставляют ресурсы и поддержку разработчикам.

Перспективный

Как мы знаем, Swift разрабатывается и поддерживается компанией Apple, и он будет основным языком для разработки приложений для iOS, macOS, watchOS и tvOS. Это даст два преимущества: мобильные приложения, созданные на Swift, обеспечат длительную производительность и сэкономят затраты на обслуживание в долгосрочной перспективе.

Взаимозаменяемость

Swift разработан для работы и сосуществования с существующим кодом Objective-C, что позволяет предприятиям интегрировать его в свои существующие приложения.

Простота использования

Благодаря чистому и лаконичному синтаксису Swift легко изучать, использовать и

читать. Это означает, что задачи программирования на Swift становятся еще легче, проще и эффективнее – благодаря поддержке замыканий и генериков, предоставляемых языком программирования.

Недостатки языка программирования Swift

Цифровая трансформация никогда не бывает без недостатков, поскольку она дает инновациям простор для совершенствования. Swift не является исключением. Их может быть много, но мы назвали три основных недостатка Swift;

1. Ограниченная поддержка старых операционных систем
2. Ограниченное количество библиотек сторонних разработчиков
3. Отсутствие кроссплатформенности, только для устройств Apple

Ограниченная поддержка старых операционных систем

Swift поддерживается только для новых версий операционной системы (ОС) Apple. Несомненно, он подходит для современной системы, но это также ограничивает его полезность для старых устройств.

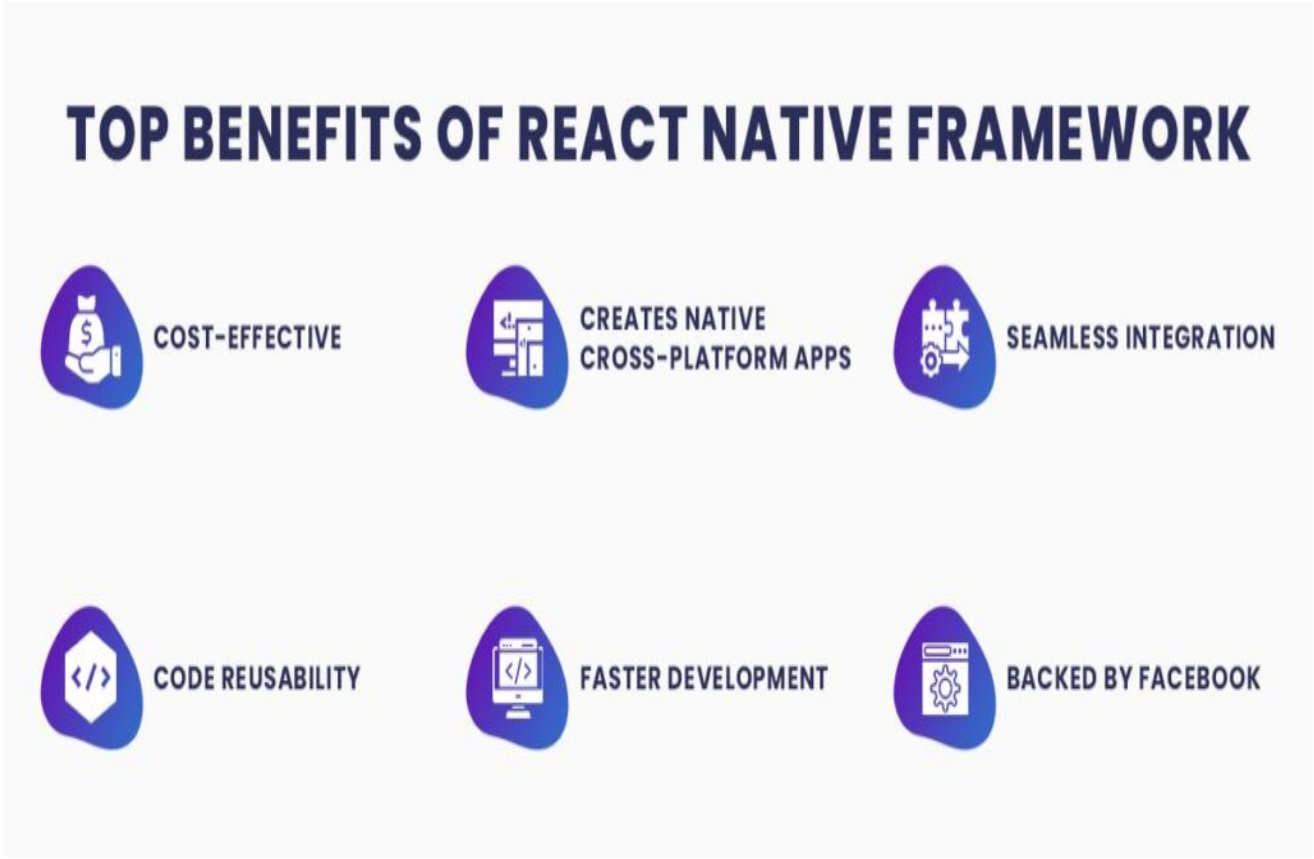
Ограниченные библиотеки сторонних разработчиков

В отличие от других языков программирования, таких как Java, Python и JavaScript, Swift поддерживает ограниченное количество сторонних библиотек. В результате некоторым Swift-приложениям будет сложно получить достаточное количество ресурсов и инструментов для лучшей работы.

Отсутствие кроссплатформенности, только для устройств Apple

Несмотря на то, что Apple называет Swift кроссплатформенным, вы не можете использовать Swift для создания приложений для Android. Но да, вы можете использовать этот язык для создания приложений для различных устройств на базе iOS.

Преимущества React Native Framework



React Native – это популярный фреймворк с открытым исходным кодом, который является лучшим выбором для создания кроссплатформенных мобильных приложений. Для разработки экономически эффективных приложений для платформ iOS и Android, React Native является наиболее предпочтительным выбором. Вот 6 лучших преимуществ использования React Native.

1. Экономически эффективный
2. Создает нативные кросс-платформенные приложения
3. Бесшовная интеграция
4. Возможность повторного использования кода
5. Быстрая разработка
6. При поддержке Facebook

Экономически эффективный

Вы можете создать стандартное приложение на React Native всего за $10k*. Несомненно, это лишь приблизительная, а не окончательная цена. Если вы работаете с разработчиком React Native в США, вам, возможно, придется потратить большую сумму, но если вы выберете такого же разработчика в Индии, вы сможете

создать приложение за четверть стоимости. Да, по сравнению с другими технологиями, это экономически выгодно, поскольку вы можете легко найти разработчиков React Native для своего проекта.

Создание нативных кросс-платформенных приложений

React Native поставляется с опцией нативного рендеринга, которая позволяет платформе создавать нативные приложения. Вот как это происходит.

```
import React from 'react';
import {Text, View} from 'react-native';
import {Header} from './Header';
import {heading} from './Typography';

const WelcomeScreen = () => (
  <View>
    <Header title="Welcome to React Native"/>
    <Text style={heading}>Step One</Text>
    <Text>
      Edit App.js to change this screen and turn it
      into your app.
    </Text>
    <Text style={heading}>See Your Changes</Text>
    <Text>
      Press Cmd + R inside the simulator to reload
      your app's code.
    </Text>
    <Text style={heading}>Debug</Text>
    <Text>
      Press Cmd + M or Shake your device to open the
      React Native Debug Menu.
    </Text>
    <Text style={heading}>Learn</Text>
    <Text>
      Read the docs to discover what to do next:
    </Text>
  </View>
```

Бесшовная интеграция

Независимо от того, хотите ли вы создать приложение с нуля или добавить его к существующим приложениям, React Native – всегда лучший выбор.

Возможность повторного использования кода

Напишите один раз и разверните его везде. Таким образом, один код React Native может быть развернут на многих платформах.

Быстрая разработка

Благодаря быстрому и эффективному фреймворку, React Native способствует более быстрой разработке мобильных приложений.

При поддержке Facebook

Поскольку фреймворк разработан и поддерживается Facebook (теперь Meta), он постоянно развивается и растет. В результате предприниматели получают новые функции и возможности для добавления в свои приложения.

Недостатки React Native

Как и преимущества, есть несколько недостатков использования React Native для разработки приложений. В основном существует три недостатка React Native. К ним относятся;

1. Сложный дизайн
2. Вялая производительность
3. Длительное тестирование и отладка

Сложный дизайн

Несмотря на то, что React Native является хорошим выбором для разработки кроссплатформенных приложений, он поставляется с проблемами дизайна, поскольку Android и iOS имеют свои собственные рекомендации. С React Native проблем нет, но это занимает больше времени, чем нужно.

Вялая производительность

Несомненно, приложения React Native работают медленнее, чем нативные приложения. Гибридные приложения проходят через дополнительный уровень абстракции. Это означает, что взаимодействие между нативными модулями и

слоями становится медленным.

Более длительное тестирование и отладка

Гибридные приложения требуют больше времени на тестирование, поскольку они требуют больше усилий и точности. Но разработка происходит быстрее по сравнению с нативными приложениями.

React Native против Swift: Сравнение возможностей

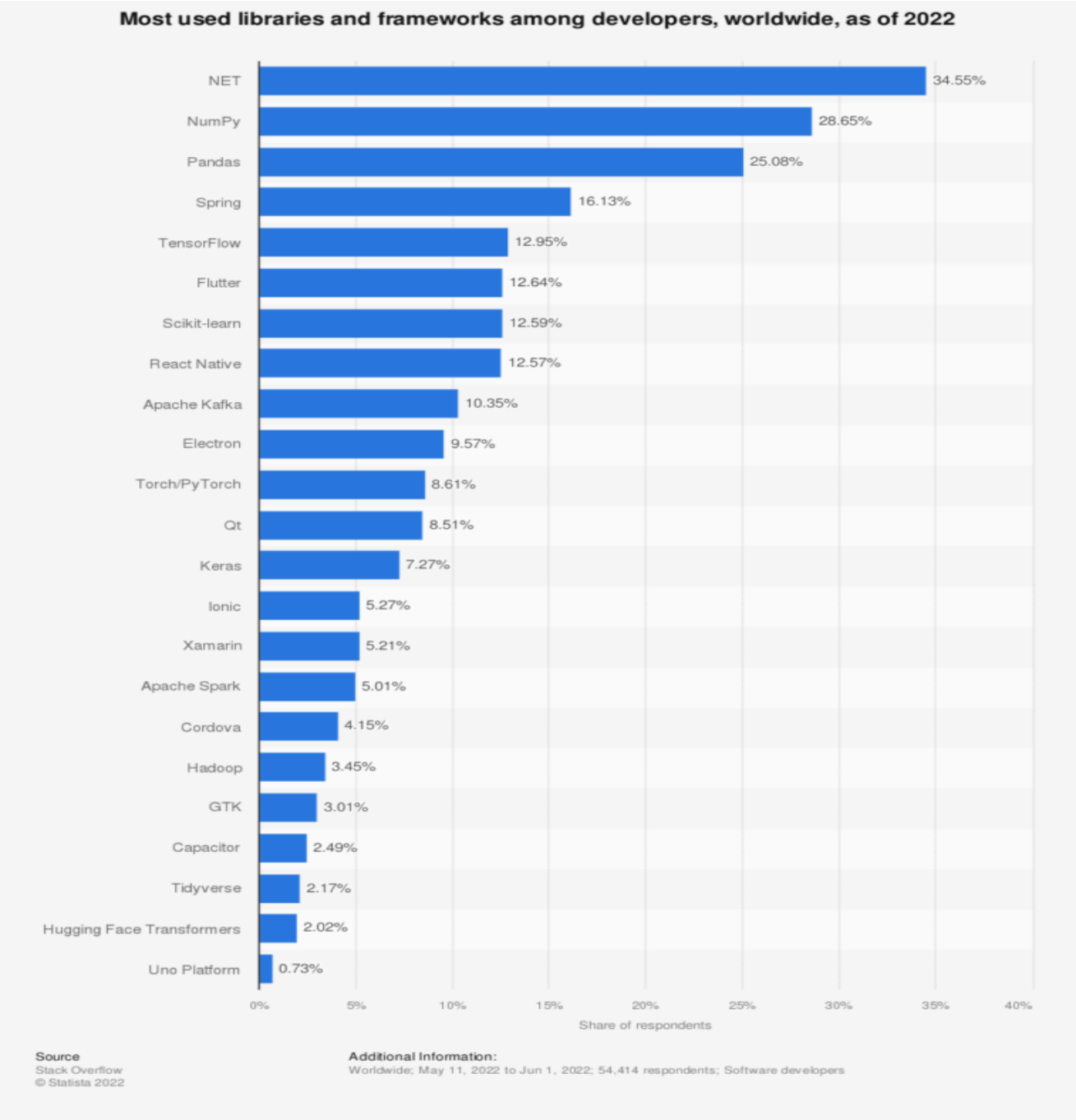
React Native и Swift – это лучшие технологические предложения для разработки приложений для iOS. Итак, давайте подойдем к финальному раунду сравнения. В этом сравнении React Native vs. Swift мы сравним следующее:

- Swift против React Native: Популярность
- Основные особенности React Native и Swift
- Топ приложений, созданных с использованием Swift и React Native
- React Native и Swift: Когда выбирать?

Итак, давайте обсудим их подробно один за другим.

Что популярнее – React Native или Swift?

Согласно данным Google Trends, популярность React Native постоянна, в то время как Swift подвержен небольшим колебаниям. Однако если говорить о языках программирования, то React Native значительно опережает Swift, его используют 65,36% разработчиков. В отличие от него, Swift известен среди 4,91% разработчиков по всему миру. Опять же, если говорить только о React Native как о библиотеке, то она занимает 8-е место по популярности: более 12,57% разработчиков приложений используют этот фреймворк.



Swift против React Native: Сравнение основных характеристик

Производительность

С точки зрения производительности React Native против Swift, нативные приложения для iOS, созданные на Swift, в целом быстрее и эффективнее, чем приложения, созданные на React Native. Несомненно, приложения React Native немного медлительны, но за последние годы их производительность значительно

выросла. Таким образом, Swift имеет преимущество перед React Native; последний все еще может обеспечить приемлемую производительность, учитывая тип приложения, которое вы создаете, и его сценарии использования.

Кривая обучения

Если вы уже знакомы с JavaScript, вы сможете быстро освоить React Native. Однако, что касается Swift, то он немного сложнее, поскольку вам необходимо с самого начала изучить язык программирования Swift.

Скорость разработки

React Native поддерживает более быструю разработку, позволяя разработчикам один раз написать код и развернуть его на платформах iOS и Android. В результате это позволяет сократить почти половину времени разработки, если вы хотите иметь мобильное приложение для обеих платформ: Android и iOS. С другой стороны, Swift специфичен для разработки приложений для нативной iOS. В результате он может быть быстрее, но только для одной платформы.

Пользовательский интерфейс

Несмотря на то, что и React Native, и Swift имеют надежные библиотеки для создания отличных мобильных приложений с отзывчивыми пользовательскими интерфейсами, SwiftUI предлагает дополнительные инструменты и фреймворки для создания лучших в своем классе пользовательских интерфейсов. Таким образом, в сравнении Swift vs. React Native по пользовательским интерфейсам (UI), оба ведут жесткую борьбу, хотя Swift имеет преимущество над своим конкурентом.

Сопровождаемость кода

React Native, похоже, выигрывает в этой гонке, поскольку он поставляется с более простым в обслуживании кодом. Разработчики React Native могут вносить изменения в кодовую базу без ущерба для производительности и поведения приложений. Swift не предоставляет такой возможности, поскольку приложения Swift требуют большего тестирования и отладки, чтобы убедиться, что изменения не влияют на общую производительность приложения.

Доступность разработчика

Вы можете нанять разработчиков React Native, не прилагая особых усилий. Однако наем разработчиков Swift может оказаться утомительным и дорогостоящим делом.

Поддержка сообщества

React Native имеет большую поддержку сообщества по сравнению со Swift. Но обе компании, несомненно, предоставляют хорошие ресурсы, включая поддержку разработчиков, документацию, форумы и учебники, и React Native сохраняет лидерство.

Стабильность

Swift является продуктом Apple и предпочтительным выбором для создания нативных приложений для iOS. В то время как React Native, несмотря на то, что справляется с тяжелыми задачами, не предпочитается разработчиками для создания нативных приложений;

Зрелость платформы

В войне React Native против Swift, последний является более зрелым, чем первый. Swift был выпущен в 2014 году, а React Native – в 2015. Поэтому, конечно, Swift сохраняет здесь лидерство.

Лучшие приложения, созданные с использованием React Native и Swift

React Native и Swift популярны среди топ-гигантов. Существует так много приложений, созданных с использованием React Native и Swift, что мы перечислили их в списке лучших.

Swift	React Native
Twitter	Instagram
Test Center	Walmart
SlideShare	UberEats

Facebook	Airbnb
Uber	Wix
Slack	SoundCloud
Accenture	Tesla
Khan Academy	Bloomberg
Lyft	
LinkedIn	
WhatsApp	

React Native и Swift: Когда выбирать?

Когда использовать React Native?

Вы должны использовать React Native, если ваш проект требует следующего:

- Приложения, требующие быстрой разработки
- Мобильные приложения, которые должны быть разработаны для платформ iOS и Android
- Приложения, которые должны быть высоко отзывчивыми
- Приложения, которые должны быть высоко настраиваемыми
- Приложения, которые требуют интеграции с веб-технологиями
- Приложения, которые требуют частых обновлений.

Например, React Native – лучший выбор для приложений для социальных сетей, электронной коммерции, игр, образовательных приложений и приложений для повышения производительности.

Когда использовать Swift?

Вы можете использовать Swift, если для вашего проекта требуется следующее:

- Нативные приложения для iOS и macOS
- Приложения, требующие высокой производительности
- Приложения, которые должны взаимодействовать с оборудованием
- Приложения, которые требуют сложных алгоритмов
- Приложения, которые требуют повышенной безопасности
- Приложения, которые требуют интеграции с существующим кодом iOS/macOS.

С небольшой разницей, Swift также можно использовать для создания приложений для социальных сетей, фитнес-приложений, приложений для электронной коммерции и приложений для повышения производительности.

Сравнение React Native и Swift с первого взгляда

Категория	**React Native**	**Swift**
Платформа	Кросс-платформа	Только iOS, macOS, watchOS и tvOS
Язык программирования	JavaScript	Swift
Инструменты разработки	React Native CLI, Expo CLI	Xcode
Компоненты пользовательского интерфейса	Компоненты, специфичные для конкретной платформы	UIKit
Производительность	Ниже, чем в нативных приложениях	Нативная производительность
Возможность повторного использования кода	Высокая степень повторного использования кода на разных платформах	Код должен быть переписан для других платформ
Кривая обучения	Простота освоения для разработчиков	Более сложная кривая обучения для новичков
Поддержка сообщества	Большое и активное сообщество	Большое и активное сообщество
Время разработки	Ускоренное время разработки	Более длительное время разработки

Отладка	Простота отладки с помощью средств разработки веб-приложений	Отладка может быть более сложной
Утверждение в магазине	Приложения подчиняются правилам магазина приложений	Отсутствие ограничений для приложений macOS
Библиотеки сторонних производителей	Библиотеки сторонних производителей	Меньше доступных библиотек сторонних производителей
Стоимость	Снижение затрат на разработку	Более высокие затраты на разработку

React Native против Swift: Кто победитель?

Оба они кажутся победителями, поскольку предлагают выдающиеся результаты, учитывая подходящий проект, для которого они использовались. То же самое мы бы посоветовали и клиентам. В некоторых случаях мы видим, что Swift имеет преимущество над React Native, в то время как в других случаях React Native вырывается вперед. Но это также факт, что один размер не подходит для всех. Будь то Swift или React Native, выбирайте технологию, исходя из требований вашего проекта. Кроме того, если вы не так хорошо разбираетесь в технологиях, найдите разработчиков приложений или компании по разработке мобильных приложений, обсудите свой проект и получите консультацию. Кроме того, сравнение нативных и кроссплатформенных технологий, а также сравнение конкретных технологий, например, React Native против Swift, также поможет вам определить, какая технология подходит для вашего проекта. Надеюсь, это вам поможет.

Дата Создания
12.05.2023