Mobile cross platform framework analysis

Student: Lukas Vismantas

Class: WU-F18v

Program: Web Development

Supervisor: Jes Arbov

Institution: Business Academy Aarhus

Introduction

As of currently, 2019 May, Android owns about 75% of the market for smartphone operating systems while ios owns most of the remaining share at 22%[1]. Making an application solely for one of these platforms means missing out on a large audience. Although native applications are naturally the most performant and intuitive, choosing native development can be expensive as it would mean having to maintain two separate codebases. It is a valid option for large companies where any increase in the usage of the application can yield profits or well funded companies which have the capital to invest. However for startup companies who do not possess extensive funding and want to break into the app market as quickly as possible it is simply not an option.

This is where cross platform development frameworks come in. They are the answer for the disjointed native technology stack problem. It is a great business solution as a single codebase to maintain means a huge cut in costs and overhead. Moreover some frameworks allow development using web technologies which greatly reduce the learning curve for someone who wishes to start developing an application but only has prior web development knowledge.

With all the benefits it provides it seems like using a cross platform development framework would be an obvious choice. That is far from the truth. With new trends and frameworks constantly emerging it can be hard to understand and find research on which technologies are right for a given use case, how they compare with each other, how flexible they are and do they scale well. Realizing you have chosen the wrong framework mid development can come at a detrimental cost.

Compared to web and native, cross platform development is a relatively new concept. However it is not an entirely different paradigm. In a way cross platform frameworks try to act as a bridge between web and native. Borrowing traits from both. With this thesis I aim to explore cross platform frameworks from the standpoint of web and native development. By doing so I hope to 1) outline the risks and the benefits that come when working with them by providing a high level overview and 2) give an in-depth look into the most promising frameworks.

[1] "mobile OS market share - StatCounter ...."
http://gs.statcounter.com/os-market-share/mobile/worldwide. Accessed 14 May. 2019.

Research method

In my initial analysis phase I will be researching what cross platform frameworks strive for as well as the problems they might possibly face. I believe the best way to achieve this is by trying to understand where cross platform stands in regards to native and web development. By outlining their strengths and weaknesses I aim to understand the goals and shortcomings of cross platform.

To create a more in-depth look I will be analyzing frameworks of my choice. When choosing a framework the key factors which I will be looking at are popularity within the developer community, adoption and the core technology behind them.

At this point I will be proceeding to my in-depth analysis phase during which I will be taking a more detailed look into each of my selected frameworks and their defining features. The analysis of the frameworks will rely both on empirical and theoretical research and will be structured as follows:
Introduction. With this section I am going to introduce the framework by providing a high level explanation of its core defining features.
Implementation. To better understand and familiarize myself with the development flow I will be implementing a small application using each framework. This way I aim to form my own opinions about it and provide myself with some context when researching further. Due to the complexities of said frameworks and time restrictions I will not be creating large scale applications or benchmarks in performance. I believe that performance is tricky to measure as it differs based on the implementation, scale of the project and the chosen technology stack. Cross platform frameworks offer multiple tools and practices for application optimization which I will not be covering in my implementation.
Development speed. Development speed can be a major indication of a poorly built framework. This problem can arise from low adoption, subpar documentation, inflexible and unintuitive APIs, lack of updates, slow iteration cycles. In this section I will be discussing certain features of the framework and how they improve or undermine development speed. This will be based on my own experience during implementation as well as what I discovered during my research.
Strengths and weaknesses. In this part I will be exploring how these frameworks achieve the previously outlined goals as well as researching of possible or existing problems the frameworks posses. Besides the experience gained during the implementation, I will be looking through the provided documentation, code samples as well as articles.
Overview. Reiteration of key discoveries and comparison to other frameworks.

The conclusion will contain the summary of my findings and further comparison of the frameworks.

Implementation case

Description

The application I will be building is a simple todo application with the functionality of viewing, creating and toggling todo objects which contain an id, title and a completed boolean. For my backend I will be using the /todos endpoint of the fake REST API provided by https://jsonplaceholder.typicode.com/.

Goal

The aim of these implementation is not the resulting application but rather for me to familiarize myself with the technologies and workflow of the chosen framework. By doing this I hope to gain context required to better understand research that is to follow afterwards as well as identify any pitfalls and form my own opinion of the framework. When explaining the implementation I will not be going into details as this is out of scope of this thesis. Instead I will give a brief overview of what is needed for set up, explanation of the files generated and a brief explanation of what I implemented in the application. As mentioned before this test case is in no way supposed to reflect any performance or other benchmarks.

Testing

I will be testing my application on the following virtual devices: Nexus 5x API 28, Pixel 2 API 26 as well as my Sony Xperia Z5 compact phone which has an API level of 25.

I will not be testing my applications on any ios devices for two following reasons. 1) I have never previously used any ios devices. As I do not have previous experience with ios devices it would be hard for me to assess whether the cross platform application behaves any differently without having a native counterpart. 2) Testing of different devices irrelevant to the goal of my implementation since I am more interested in the development process rather than the resulting application.

Analysis

Web vs Native vs Cross platform

Web application development

Pros:

Easy to learn. They are built using regular web standards - html, js and css.

Easy to maintain. Users do not need to install updates or visit the app store since the newest version is always what loads when a user accesses a web app. They are hosted as any regular website would be.

Use with any technology or language. There are no limitations to what web technology stack you want to use.

Single codebase for all platforms. Will run on any device that can open a browser.

Cheap. It is easier to find a web developer than a native mobile developer. In the stackoverflow 2019 survey of 90,000 participants only 18.1% participants identified as mobile developers compared to backend - 50.0%, fullstack - 51.9% or front-end 32.8%.[2]

Cons:

Needs a browser to run. Limited by the rendering engine of the browser.

Slower than native apps. Web applications (unlike native ones) are not built to be fully optimized for a specific device.

Less interactive and less intuitive.You need to invest extra work to make the application feel more native like.

Cannot interact with device utilities . They do not have direct access to all native APIs.

[2] "Stack Overflow Developer Survey 2019 - Stack Overflow Insights." https://insights.stackoverflow.com/survey/2019.

Native application development

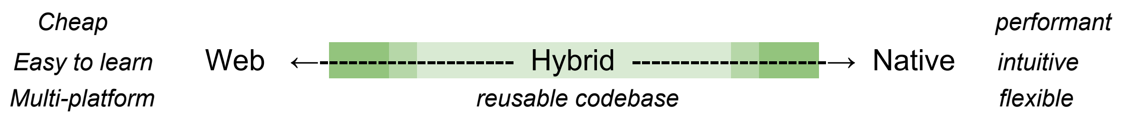
Pros:
Fast.They are built for that specific platform making them as performant as possible.
Interactive and intuitive by default.
Full access to the device capabilities.

Cons:
Single platform. Developing an application with Kotlin/Java will only work for android and developing with Objective-C/Swift will only work on ios and if you require web presence you need a third codebase.
Huge learning curve. There are not as many native developers as web developers. Native would come with bigger learning curve as it is written in strictly typed languages with complex architecture. Having a developer work on both platforms would require knowledge of both ios and Android.
Expensive. As the app scales maintaining two codebases can create massive overhead.

Cross platform application development

The core reason why cross development frameworks exist is that they aim to create a reusable codebase to reduce development costs. However they also aim to reduce the overhead needed to maintain that one codebase. As you can see some of the biggest drawbacks of native development are the biggest pros of web development. Cross platform application development is trying to find its niche somewhere in between. It is striving for the best traits of both paradigms.

Performance and flexibility of the native. Naturally it is preferred for the application to be as close to in performance to a native application. Flexibility refers how easy it is to access all the API's that the target platform supports.
Learning curve and cost of the web. As mentioned before web development comes with a smaller learning curve due to it being adopted much more widely. Some frameworks take advantage of this by incorporating weblike technologies into their development flow.



However in their stretch to achieve all this I believe they are likely to face the following **negative sides:**

Underperformance. Applications developed can vary in how native-like they are. Even frameworks that market themselves as creating fully native applications do not necessarily mean they can be as performant natively developed applications. This will most likely lead to slower performance than Native.
Inflexibility. Poor abstractions of native API can lead to slow development times and applications that feel unintuitive compared to their native counterparts.
Low adoption. Cross platforms frameworks are still new in the market compared to native app development. Widely adopted and company backed frameworks are generally safer. They are more likely to have faster update cycles, better documentation and more resources online. The community ecosystem is a major factor to consider when choosing any technology.
Dependencies. As these frameworks try to bridge web and native they might be relying on external libraries to either recompile code, run debugging servers or create a smoother development flow. They are prone to cause problems by design. The more dependencies used, the harder it is to debug problems as they might arise from said libraries. It also creates a higher learning curve as there are multiple sources of knowledge about what is used in your project.[3]

[3] "Dependency hell - Wikipedia." https://en.wikipedia.org/wiki/Dependency_hell.

Chosen Frameworks

React Native
Why I chose it
Github stars 75K[4]
Backed by Facebook and used for creation of user heavy and already established applications like facebook, instagram, pinterest, uber, discord[5]. React Native is the most widely accepted and used framework for creating cross platform compiled native applications using a javascript bridge. In the 2019 Stackoverflow survey 10.5% of participants responded to using React Native placing it as the most popular cross platform framework. It combines its development flow with the frontend framework *React*. In the same survey, React placed as the second most popular framework with 31% of the participants having used it.[6]

Flutter
Why I chose it:
Github stars 58K[7]
This framework has as an unique implementation for creating native application using a 2D graphics library called Skia. It is used by applications like Google Ads, Tencent and Alibaba[8]. Even though it is still new in the market I feel it is important to review it not only due to its innovative approach but also to it being backed by *Google*. Having a huge company backing your framework gives it an edge over its smaller competitors (Nativescript, Cordova) and leads to better developer experience(faster and more quality updates). *Google* also owns the *Android* OS. If this is a break into the market they have many ways to back *Flutter*'s success. *Flutter* has a rapidly growing community following[9]. In the 2019 Stackoverflow survey 3.4% responded to having used Flutter and it placed as the third most loved technology with 75.4% respondents reporting having loved working with it.

Apache cordova
Github stars 37K (Ionic)[10] *14K(Framework 7)*[11] *10K(Jquery Mobile)*[12]
Apache Cordova is the main framework used for creation of hybrid applications with the use of webview technology. Hybrid applications meaning that the applications are not fully native nor

[4] https://github.com/facebook/react-native. Accessed May. 2019.
[5] https://facebook.github.io/react-native/showcase.
[6] https://insights.stackoverflow.com/survey/2019
[7] https://github.com/flutter/flutter. Accessed May. 2019.
[8] https://flutter.dev/showcase
[9] https://star-history.t9t.io/#facebook/react-native&facebook/react-native&flutter/flutter Accessed May 2019.
[10] https://github.com/ionic-team/ionic Accessed May. 2019.
[11] https://github.com/framework7io/Framework7 Accessed May. 2019.
[12] https://github.com/jquery/jquery-mobile/ Accessed May. 2019.

are they web applications either but rather something in between. It is usually used in tandem with UI frameworks like *framework 7, jquery Mobile* or *Ionic* to create a more native experience.

In a 2019 StackOverflow survey based on fifty thousand participants 7.1% percent responded they use *Cordova* placing it as the second most popular cross platform mobile development framework.

Notable mentions

Nativescript

Github stars 16K[13]

Nativescript much like React Native implements a javascript bridge to communicate with the native API. *Nativescript* supports *Angular* and *Vue.js* for its frontend. It also offers a different threading model than that of React Native and its frontend is written in HTML like XML structures. It I did not chose *Nativescript* due to its low adoption[14].

Xamarin

Github stars 3K[15][16]

Owned by Microsoft. Uses a C# codebase and the .NET framework for developing for all platforms by compiling to native code. The UI is designed separately for each platform. However using Xamarin.Forms it creates possibility for reusable code between platforms while also providing tools like Xamarin.Ios and Xamarin.Android for further customization. Big upside of Xamarin is that it is strongly typed to the native bindings thus resulting in a more robust code.[17] It is most definitely a valid option for someone coming from a strongly typed language background.

In the 2019 StackOverflow survey 6.5% participants responded that they use Xamarin (there has been a drop from 7.4%[18]), 51.7% of the same developers reported not enjoying working with it. Since C# is not as popular as web based languages and due to the fact that UI is platform specific it is most likely to be more expensive to produce Xamarin applications.

[13] https://github.com/NativeScript/NativeScript Accessed May. 2019

[14] https://star-history.t9t.io/#NativeScript/NativeScript&NativeScript/NativeScript&facebook/react-native Accessed May. 2019

[15] Such a low Xamarin result can be explained by the fact that it has only recently become an open source product and still requires proprietary Visual Studio IDE to be used

[16] https://github.com/xamarin/Xamarin.Forms Accessed May. 2019.

[17] https://docs.microsoft.com/en-us/xamarin

[18] https://insights.stackoverflow.com/survey/2018

React Native

Introduction

Developed by facebook and open-sourced in March 2015.[19] React Native lets you build native mobile apps using javascript. It uses React for its frontend framework. The UI is written in JSX using custom XML components which get translated to native UI components using a javascript bridge.

Javascript bridge

You can think of a javascript bridge as a layer which translates your javascript code to native commands. There is a total of 4 threads running within a react native application[20][21]. Our point of interest is the:
UI thread. Runs in each standard native app. It handles displaying the elements of the user interface and processes user gestures.
Javascript thread. Its task is to execute the JavaScript code in a separate JavaScript engine (Webkit on ios, V8 on android).

These two threads never communicate directly and never block each other. Between these two threads is the so-called bridge which is the core of React Native. The bridge is built in C++ meaning it can be run on multiple platforms and it has three important characteristics:

Asynchronous. It enables asynchronous communication between the threads. This ensures that they never block each other.

Batched. It transfers messages from one thread to the other in an optimised way.

Serializable. The two threads never share or operate with the same data. Instead, they exchange serialized messages.[22] [23]

$$\begin{tikzcd}
\text{Main thread} \arrow[r, shift left] & \text{Bridge} \arrow[l, shift left] \arrow[r, shift right] & \text{Javascript VM} \arrow[l, shift right]
\end{tikzcd}$$

[19] "React Native: Bringing modern web techniques to mobile - Facebook ...." 26 Mar. 2015, https://code.fb.com/android/react-native-bringing-modern-web-techniques-to-mobile/
[20] "React Native App how many threads? Execution Model ... - WebKJ." https://www.webkj.com/react-native/react-native-app-how-many-threads.
[21] Besides **UI Thread** and the **Javascript** thread there is also the
Native Modules Thread This threads handles the api access to platform API.
Render Thread - React Native render thread is used to generate OpenGL commands used to draw your UI.
[22] "How React Native Works ? | Codementor." 16 Aug. 2018, https://www.codementor.io/saketkumar95/how-react-native-works-mhjo4k6f3.
[23] "React Native: What it is and how it works – We Talk IT – Medium." 12 Jun. 2017, https://medium.com/we-talk-it/react-native-what-it-is-and-how-it-works-e2182d008f5e

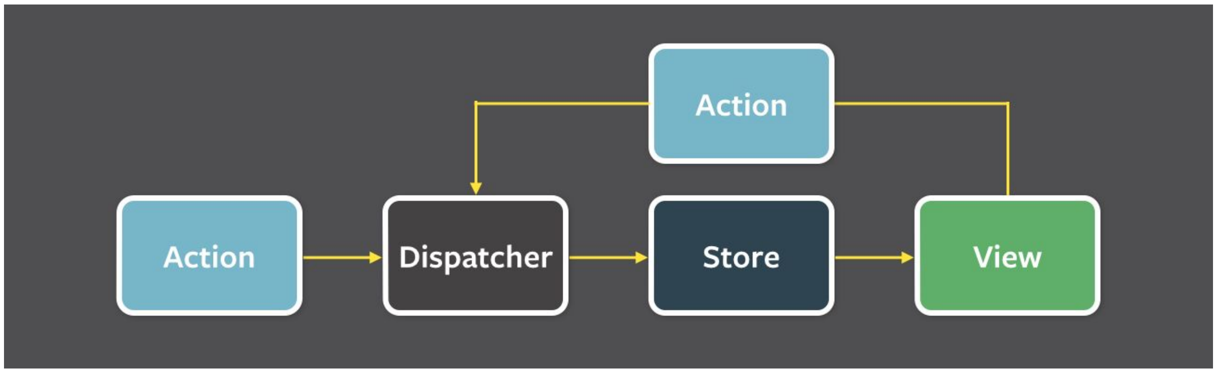
React

React is a JavaScript library open-sourced back in 2013 by Facebook. It is made to improve the the development of scalable single page web applications. Unlike larger MVC-style JavaScript frameworks (Angular, Ember, Backbone), React is very modular in its approach to additional functionality like routing, state management or interaction with it's API[24] (similar to Vue.js). It is the 5th most starred repository on Github[25]. According to the stack overflow 2019 survey it is also the second most used technology.

Flux design pattern

In the facebook docs it is stated that:

“Flux eschews MVC in favor of a unidirectional data flow. When a user interacts with a React view, the view propagates an action through a central dispatcher, to the various stores that hold the application's data and business logic, which updates all of the views that are affected. This works especially well with React's declarative programming style, which allows the store to send updates without specifying how to transition views between states.” [26]



Credit to Tim Ermilov [27]

Store layer. Singleton services where state is stored. It makes sure it is synchronized. Components can have specific data but if you have data which is app wide(multiple aspects of your app use and rely on your data) it is meant to be stored within the store. The data from the store gets passed down to components

View layer. Where UI components and the business logic is stored. Stores data chaning is what triggers the changes in the view removing the need to communicate with any controllers.

Actions. Commands sent to the dispatcher which in turn update the store. Meant to be simple javascript objects which only contain the name of the action and the data.

Dispatcher. Singleton services that in all the actions and send them to the store.

[24] https://en.wikipedia.org/wiki/React_(JavaScript_library)

[25] https://github.com/search?p=1&q=stars%3A%3E1&ref=searchresults&type=Repositories&utf8=%E2%9C%93 Accessed May. 2019

[26] https://facebook.github.io/flux/docs/in-depth-overview.html

[27] "Using Postal.js as Flux Dispatcher in your modular React application ...." 24 Jun. 2015, https://towardsdatascience.com/using-postal-js-as-flux-dispatcher-in-your-modular-react-application-with-turris-js-1355e9feacc9.

Native UI

To employ the reactive part of React a VDOM is being used[28]. However they took the concept of VDOM a step further and made it render it to native components instead of browser DOM. This allows the use of Hot reloading. The UI and styles are written using JSX and then are translated to native UI and styles.

JSX

Some frameworks, like Nativescript, use a special templating language which lets you embed code inside markup language. In React, this is reversed. JSX lets you write your markup language inside code. It allows the use of HTML/XML structures to define UI components and style. During compilation they get translated to their native equivalents.

Implementation

Source code
https://github.com/Louvki/react-native-todo

Set up (windows)
There are two ways to set up a React Native project.
One of them is using a neat tool called Expo. To set it up all you are required to do is install the expo-cli. From there on expo will set up a development server for your project which you can access by installing the expo app on the phone and connecting to the same wireless network as your computer. It is great for prototyping and just trying out React Native however because you do not build any native code when using Expo to create a project, it is not possible to include custom native modules beyond the React Native APIs and components that are available in the Expo client app. So it is not an option for building large scale applications.

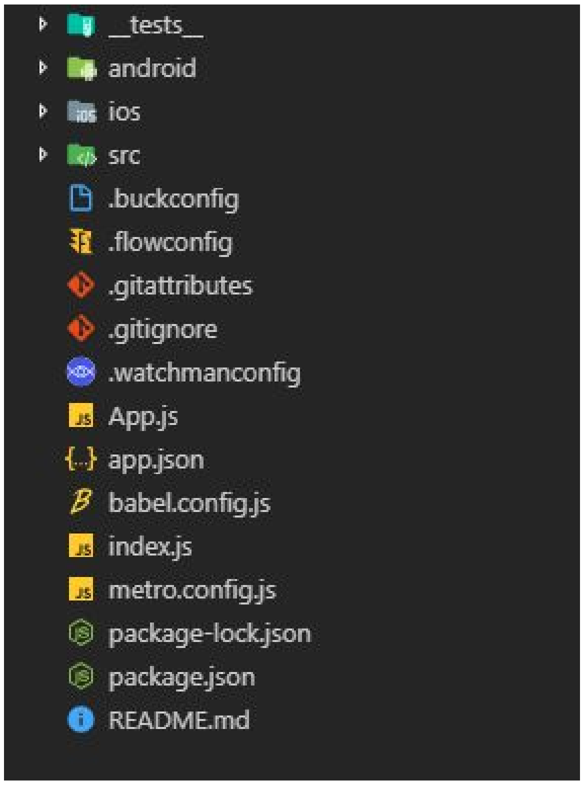
If you want to set up a development server on your machine you will be required to install Node, Python2 and JDK. To set up the environment for android development you also are required to get the Android Studio and with it the Android SDK. Afterwards you need to set up the ANDROID_HOME environment variable. You are also required to add the platform tools to the Path variable.

I installed Node, Python2 and JDK using chocolatey package manager. To aid me in my development I also installed the React Native Tools plugin for Visual Studio Code. I used the React Native CLI tool to generate a new project with the command:

```
react-native init ProjectName
```

[28] VDOM (a virtual representation of the real DOM) which gets synced with the “real” DOM. This way you can tell the UI which state you want it to be in and the DOM will match that state automatically with the use of data binding.

Generated file structure



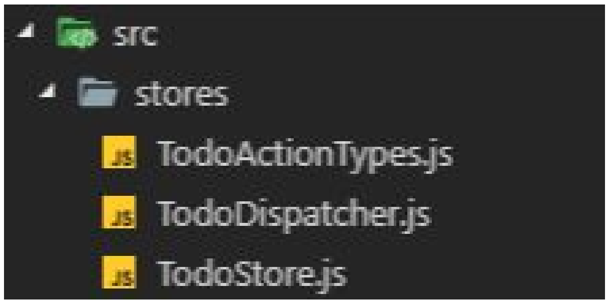
```
▸ __tests__
▸ android
▸ ios
▸ src
  .buckconfig
  .flowconfig
  .gitattributes
  .gitignore
  .watchmanconfig
  App.js
  app.json
  babel.config.js
  index.js
  metro.config.js
  package-lock.json
  package.json
  README.md
```

tests - Testing related files.
android - Android specific files like the AndroidManifest.xml, build.gradle etc.
ios - Ios specific files like Info.plist
src - This is where you place your source code.
App.js - entry react component
App.json - Application related settings, like the name.
Babel.config.js - Configuration for babel. Babel lets you convert ECMAScript 2015+ code into a backwards compatible version of JavaScript.
index.js - entry point to the application
Metro.config.js - Configuration file for the metro development server on which your application is ran.
Package.json - For managing dependencies and any other related metadata

Implementation
To better understand how Flux works and what are the benefits of it I implemented it in my project.

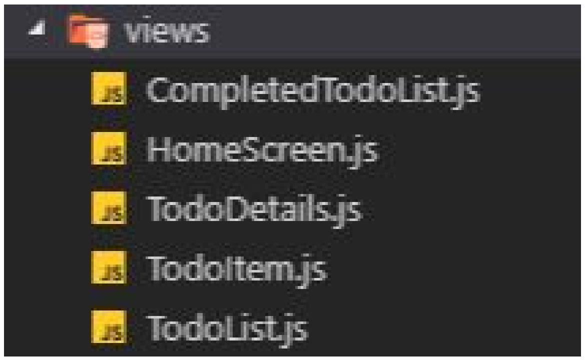
Dependencies used:

```
  "flux": "^3.1.3",
  "react": "16.8.3",
  "react-native": "0.59.5",
  "react-native-elements": "^1.1.0",
  "react-native-gesture-handler": "^1.1.0",
  "react-native-vector-icons": "^6.4.2",
  "react-navigation": "^3.8.1"
```



```
src
  stores
    TodoActionTypes.js
    TodoDispatcher.js
    TodoStore.js
```

I only created one Todo store which would be accessible globally within the application. The **TodoActionTypes.js** is enum which describes what kind of actions can be sent to the **TodoDispatcher.js**. The two action types I have is Complete_todo and Add_todo. These actions mutate the data within the **TodoStore.js**. The views listen to the data changes within the and update accordingly



```
views
  CompletedTodoList.js
  HomeScreen.js
  TodoDetails.js
  TodoItem.js
  TodoList.js
```

CompletedTodoList.js - Contains all the completed todos.
HomeScreen.js controls which component to display.
TodoDetails.js is the screen for adding a Todo to the store.
TodoList.js is where all active todos are displayed.
TodoItem.js is the component tasked with the displaying the todo.

Development speed

Hot reloading[29]. You can see the changes happen to your app in real time without losing state. It allows the developers spend less time on building and re-building the application.
JSX[30]. By allowing you to write styles, UI and code in the same file it greatly reduces file switching overhead.
React. A great front end framework which makes implementing routing, data binding and state management a lot easier.
Large community. There is already a large and still growing community surrounding React. Makes troubleshooting problems easier.

Web technologies. You can make your app work with any javascript library however you are still limited to having to use the React frontend framework. Since the UI components and style get translated to Native UI naturally you do not have as much freedom of choice as you are limited by the underlying Native UI.
Javascript. Javascript is an untyped language which can lead to runtime errors and hard to debug code as the application scales[31]. It is always possible to use something like Typescript to negate this problem. However although it is supported it is not necessarily integrated into React Native.

Strengths and weaknesses

The use of the React framework combined with JSX makes the experience of developing UI a lot easier by eliminating file switching overhead and boilerplate. JSX is also meant to run faster than vanilla javascript [32]. Using Native components also allow integrating React Native into an existing application. This allows for flexibility as your app can have Native and React Native parts.

However despite being native, its performance is far from it. When excessively using the javascript bridge bottlenecks start to occur[33]. There are ways to minimize the problem[34] by optimizing how your application works. However there will be a need to rely on writing platform specific code to make your app as performant as possible. Do not be discouraged though, when

[29] "Introducing Hot Reloading · React Native - Facebook Open Source." 24 Mar. 2016, https://facebook.github.io/react-native/blog/2016/03/24/introducing-hot-reloading.
[30] "Introducing JSX – React." https://reactjs.org/docs/introducing-jsx.html.
[31] "React Native at Airbnb: The Technology – Airbnb Engineering - Medium." 19 Jun. 2018, https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838
[32] "JSX - a faster, safer, easier JavaScript." https://jsx.github.io/.
[33] "Performance Limitations of React Native and How to Overcome Them." 19 Jun. 2016, https://medium.com/@talkol/performance-limitations-of-react-native-and-how-to-overcome-them-947630d7f440
[34] "6 Tips You Want to Know About React Native Performance — Simply ...." 7 Jun. 2017, https://www.simplytechnologies.net/blog/2017/6/6/6-tips-you-want-to-know-about-react-native-performance

optimized the apps behave nativelike. Airbnb in their review of React Native stated stated that “in practice, this was rarely a problem. Most of our React Native screens feel as fluid as our native ones” [35].

There are some unavoidable problems with initialization times. When your application loads it needs to initialize the javascript bridge which slows it down significantly, Even though you can minimize the problem with lazy loading and caching it is still an issue. AirBnB has stated that: “Before React Native can render for the first time, you must initialize its runtime. Unfortunately, this takes several seconds for an app of our size, even on a high-end device. This made using React Native for launch screens nearly impossible.”[36]

In the end React still is limited by the underlying APIs. In the same article it was also stated that it was an issue as the gesture system makes it hard to come up with unified API for both Android and ios. Performance wise React Native lists are not as performant as Native alternatives (or even Nativescript)[37]. However React Native already addressed this issue and are working on redesigning their threading model[38].

Learning curve

JSX Using the documentation I had no problem finding which components to use and JSX is very much like html.
React. If you have not used any frontend frameworks like Angular or Vue.js you will need to familiarize with concepts like props, data-binding, routing and component nesting. This can come at a pretty steep initial curve. Luckily React has extensive documentation and a large community surrounding it.

Overall the development although not purely weblike can seem very similar to it.
Flux Might be a bit harder to wrap your head if you are not familiar with software design patterns. However you are not required to enforce any kind of pattern and React Native also supports the use of patterns like Redux, Mobx or Rxjs.

[35] "React Native at Airbnb: The Technology – Airbnb Engineering - Medium." 19 Jun. 2018, https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838.
[36] React Native at Airbnb: The Technology – Airbnb Engineering - Medium." 19 Jun. 2018, https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838.
[37] "Would Airbnb Have Fared Better With NativeScript Instead of React ...." 27 Jun. 2018, https://www.nativescript.org/blog/would-airbnb-have-fared-better-with-nativescript-instead-of-react-native
[38] "State of React Native 2018 - Facebook Open Source." 14 Jun. 2018, https://facebook.github.io/react-native/blog/2018/06/14/state-of-react-native-2018. Accessed 14 May. 2019.

Overview

The UI is completely native making it the most intuitive as it can be. Meanwhile JSX makes the overall development flow feel a bit more weblike. It does come with the learning curve of a frontend UI framework React. Hot reloading is also a great feature which allows to spend less time building and more time coding, but compared to Flutters hot reload and Ionics live reload this one was the slowest.

In performance, React Native stays behind an native application and relies on various optimizations to feel more native like especially when considering initialization times. They are facing some issues due to their threading model. It is still faster and more intuitive than a hybrid application (Cordova Apache) as it is run natively and relies on native UI. This likely reduces update lag as you do not need to imitate native UI.

React being a native application allows extensive access to the underlying Native APIs. However it seems like it is facing an issue of native APIs being too difficult to abstract in a non constraining way. Airbnb has stated that although it is not as performant as native when optimized it can feel native like. This is achieved by extending the app with native code and native views[39]. In a way it allows for flexibility and better performance but at the cost of code-reuse as this does come with having to manage more native specific code and it might be a problem as your application scales.

By comparing the StackOverflow survey results from this and previous years it is clear that React Native as well as React are still widely adopted and loved frameworks and this project will be continued to be maintained. This means even more packages and support.

All in all React Native, compared to other cross platform frameworks, is a mature stable and generally well received framework. Its choice to use Native UI allows for a native experience however at the cost of code-reuse. It is a rather big tradeoff to make if you are considering scaling your application.

[39] "Pros and cons of React Native - mobile apps with JavaScript · Devbridge." 28 Apr. 2016, https://www.devbridge.com/articles/pros-cons-of-react-native-crash-course/.

Flutter

Introduction

Developed by *Google* and released to Beta on February 27, 2018[40]. As of this point in time Flutter is the latest and fastest growing technology in the mobile development field. It brings a lot of innovation to the field with its use of 2D graphics library Skia. It also strays from web paradigms by using dart instead of javascript and widgets instead of XML type UI.

Dart

Flutter applications are written in the dart programming language. It is an object oriented, class defined, dynamically typed language[41]. It uses AOT (ahead of time) compilation to compile to machine code (native instructions)[42].

Canvas UI

Instead using wrapping Android/ios specific UI components (which is what React Native, Xamarin and Nativescript does) or a displaying the UI using html (Cordova Apache) Flutter draws the UI from scratch on a screen canvas using 2D graphics library called Skia[43][44]. So buttons, text, media elements, background are all drawn by Flutter's graphics engine.

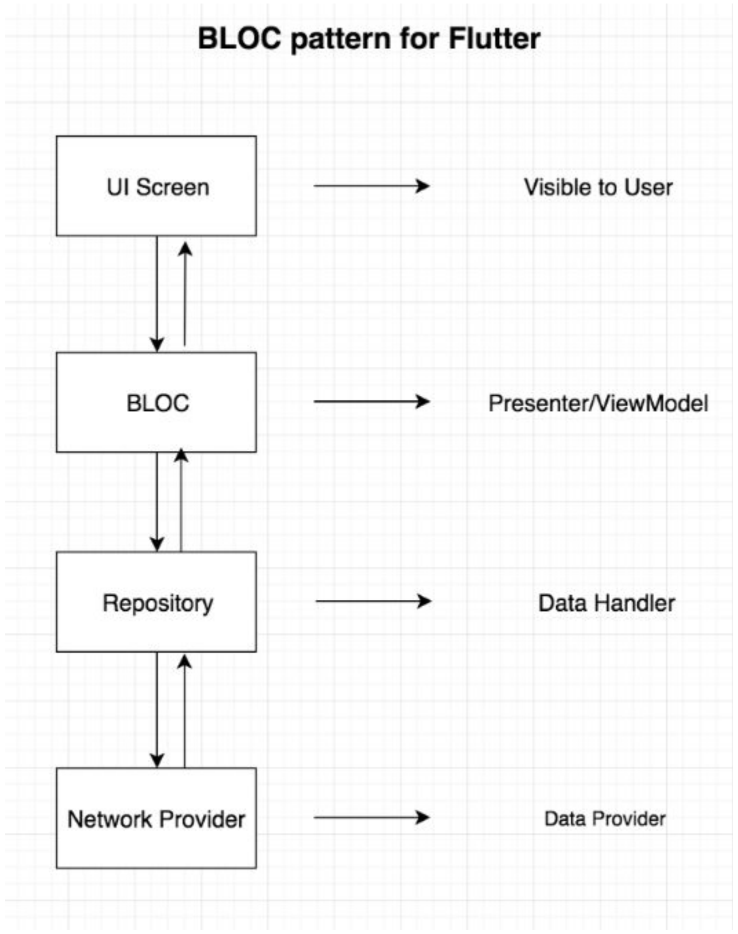
Widgets

Flutter takes a step back from markup languages and introduces widgets.They are used for everything from UI elements, styles, themes, to even state management. Due to this UI is written as code and there is no longer any difference between the code and the UI.

Bloc pattern

You can use redux, mobx or even Flux however Flutter encourages the use of the BloC pattern [45] It is heavily inspired by Flux. BLoC stands for Business Logic Component and is a state management pattern[46]. It is based on functional programming and is similar to the Rx observer pattern. Bloc is built on top of RxDart; however, it abstracts all of the RxDart specific implementation details. In order to use Bloc, it is important to have at least some understanding of Streams and how they work.

[40] https://github.com/flutter/flutter/releases/tag/v0.0.6
[41] "Language Tour | Dart." https://dart.dev/guides/language/language-tour
[42] "FAQ - Flutter." https://flutter.dev/docs/resources/faq. Accessed 14 May. 2019.
[43] "FAQ - Flutter." https://flutter.dev/docs/resources/faq. Accessed 14 May. 2019.
[44] "Skia Graphics Library." https://skia.org/. Accessed 14 May. 2019.
[45] "List of state management approaches - Flutter." https://flutter.dev/docs/development/data-and-backend/state-mgmt/options..
[46] "BLoC design pattern with Angular – Iacolaco-blog – Medium." 23 May. 2018, https://medium.com/lacolaco-blog/bloc-design-pattern-with-angular-1c2f0339f6a3.



Credit: Sagar Suri [47]

The BLoC pattern achieves a reactive approach for handling data with the use of streams. BLOCs are implemented with “Sinks” and “Streams”. When data is passed down to a BLOC via a sink, the BLOC performs some sort of business logic on the data and passes it down via a stream. So even though it has unidirectional data flow, separation of concerns between layers is still enforced. This encapsulates the business logic within the BLOC layer making it reusable in different contexts.[48]

[47] "Architect your Flutter project using BLOC pattern – FlutterPub – Medium." 26 Aug. 2018, https://medium.com/flutterpub/architecting-your-flutter-project-bd04e144a8f1.
[48] "Flutter - Reactive Programming - Streams - BLoC - Didier Boelens." 20 Aug. 2018, https://www.didierboelens.com/2018/08/reactive-programming---streams---bloc/. Accessed 14 May. 2019.

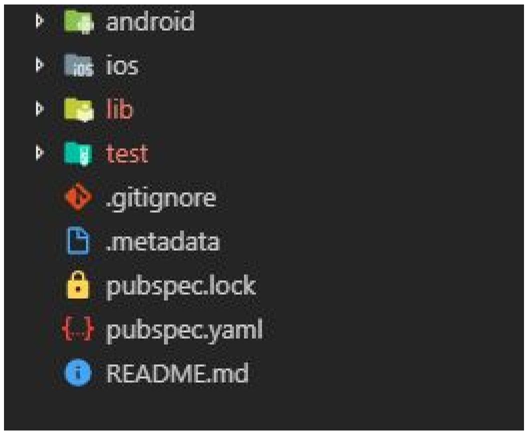
Implementation

Source code:
https://github.com/Louvki/flutter-todo

Set up:
Install Android studio (Java SDK)
Instal Flutter SDK
Update the PATH variable to able to run flutter commands from the windows console.

To improve my development experience I also installed the “Flutter” and “Dart” plugins for visual studio code (Alternatively Flutter also offers a complete, integrated IDE experience for Android Studio and IntelliJ IDEA.).

Generated file structure
Using the “flutter create” command scaffolds the following file structure.



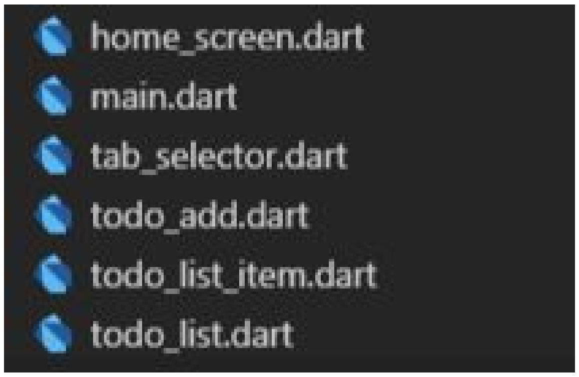
```
android
ios
lib
test
.gitignore
.metadata
pubspec.lock
pubspec.yaml
README.md
```

android - Android specific files
ios - Ios specific files.
lib - This is where you place your source code.
test - Testing related files.
pubspec.yaml - File for managing dependencies

Implementation
To better understand how BLoC works and what are the benefits I implemented it in my project.

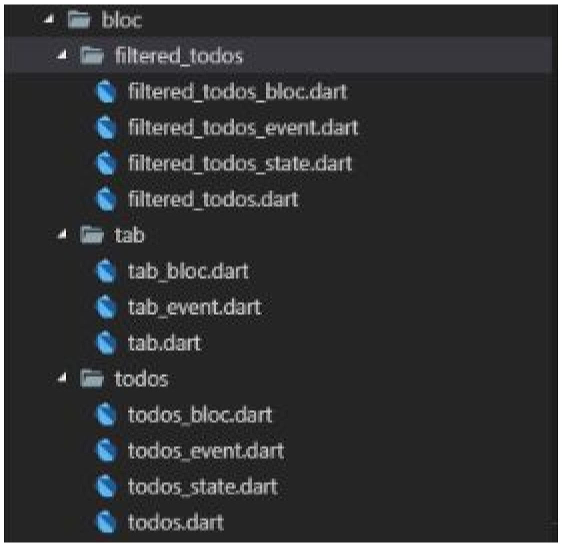
Used dependencies:

cupertino_icons: 0.1.2	
flutter_bloc: ^0.11.0	for easier BLoC implementation
http: ^0.12.0	Module for making http fetch requests
equatable: ^0.2.0	Module for equating objects.



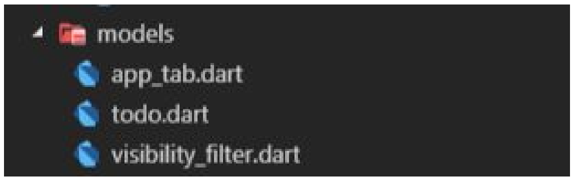
```
home_screen.dart
main.dart
tab_selector.dart
todo_add.dart
todo_list_item.dart
todo_list.dart
```

Main.dart is the entry point to our application. It has the child widget of **home_screen** which with the help of **tab_selector** handles the displaying of the tabs: **todo_list** and **todo_add. todo_list_item** is a widget used to display singular todos within the the **todo_list**



```
bloc
  filtered_todos
    filtered_todos_bloc.dart
    filtered_todos_event.dart
    filtered_todos_state.dart
    filtered_todos.dart
  tab
    tab_bloc.dart
    tab_event.dart
    tab.dart
  todos
    todos_bloc.dart
    todos_event.dart
    todos_state.dart
    todos.dart
```

Within the bloc folder I have three bocs.
tab - contains the data for the current tab/ screen we are on(active, completed or add).
todos - holds the state of all the todo objects we have fetched from our backend.
filtered_todos - Relies on the todo bloc. It contains and filters the todos based on the filter we provide.



```
models
  app_tab.dart
  todo.dart
  visibility_filter.dart
```

Since Dart is a dynamically typed language it allows the use of generics, which are great for enforcing type safety.
app_tab - enum for checking which tab we are on.
todo - Generic class for the todo objects.
visibility_filter - enum for the way we want to filter our todos. We can choose to display all, completed or active

Development speed

Hot reloading.[49]
Widgets. The switch in programming paradigm takes getting used to but widgets feel intuitive to use and greatly reduce amount of boilerplate needed for building native like components. Flutter also provides premade Material Design(Android) and Cupertino(ios) widgets as well as as a more generic fully customizable widgets.
Dart. Is a dynamically typed language (similar to Typescript). This Improves code quality by introducing type safety and generics without any overhead of having to set it up.
UI framework. Much like React Native Flutter comes with a built in UI framework.
Bloc pattern. Bloc encourages reactive programming[50] which in turn leads to more error proof code as you are less likely to mismanage state. It also seems to reduce boilerplate (compared to Redux or Mobx) as you do not have to write reducers or actions [51].

Inbuilt support for theming[52].

UI as code. Flutter does not come with a separate layout markup language. Each widget is written in Dart only, in one place, eliminating syntax and file switching overhead. However at the same time it can introduce hard to read code.
Growing community. As this technology is bleeding edge, there is no established following as compared to something like React Native. However there seems to be plentiful articles on Flutter development as well as great documentation provided by google.

Strengths and weaknesses

Since flutter is written in dart, which compiles to machine code, it eliminates the need for a javascript bridge for communicating between the native APIs/UI and your application. This in theory makes it faster than any javascript bridge based framework and it comes with reduced initialization times. In the docs they state that “Flutter aims to provide 60 frames per second (fps) performance, or 120 fps performance on devices capable of 120Hz updates”[53]. When tested it proved to be still

[49] "Hot reload - Flutter." https://flutter.dev/docs/development/tools/hot-reload
[50] "What is Reactive Programming? – Keval Patel – Medium." 12 Dec. 2016, https://medium.com/@kevalpatel2106/what-is-reactive-programming-da37c1611382.
[51] "Architect your Flutter project using BLOC pattern – FlutterPub – Medium." 26 Aug. 2018, https://medium.com/flutterpub/architecting-your-flutter-project-bd04e144a8f1.
[52] "Flutter Theme Class | Flutter By Example." https://flutterbyexample.com/flutter-theme-class/
[53] "Flutter performance profiling - Flutter." https://flutter.dev/docs/testing/ui-performance.

considerably more cpu intensive than native code on ios[54].

They state in the flutter docs that the reason why they are not using OEM widgets[55] as to not be limited by their quality or performance[56]. This approach allows the framework to bypass the hardcoded set of gestures imposed by the native systems making your code a "first-class participant in the gesture system"[57]. It also creates the tools for creating and sharing widgets which are independent of underlying OEM widgets. This can lead to easier creation and maintenance of flexible, motion based, responsive UIs[58][59].

However it does come with downside of making it harder to integrate flutter within an existing application[60]. It is possible but there are limitations since you will only be able to have one Flutter Activity, ViewController, View, etc in an application as a general rule. Also since it does not use native widgets, Flutter widgets will always be at best an imitation of the native widgets.

Currently Flutters biggest downside is that code cannot be reused for the web. However in the FAQ it is stated that the team is working on making this a possibility [61]. Another point against Flutter is low adoption. In the 2019 Stack Overflow survey only 3.4% of responded to having used Flutter. So finding a developer who already knows dart or Flutter can be quite a hard task. However it does seem like the community for Flutter is quickly growing.

Learning curve

Dart Uses C-style syntax and follows all the ECMA standards making it feel similar to Javascript. It is class-based and optionally typed, it also has asynchronous libraries to support development using streams and futures(promises). Although it is a new language any web developer should have no problem picking it up as it is very similar to Typescript and Javascript.
Widgets. To build UI you will need to learn about widgets. Flutter makes it easier by allowing you to use premade google's material design and Cupertino widgets. However despite that it still comes with a whole different paradigm of coding.

What is nice about Flutter is it does not use any outside dependencies - just the flutter

[54] "How fast is Flutter? I built a stopwatch app to find out. - freeCodeCamp ...." 18 Mar. 2018, https://medium.freecodecamp.org/how-fast-is-flutter-i-built-a-stopwatch-app-to-find-out-9956fa0e40bd.
[55] OEM - original equipment manufacturer. Pretty much meaning native UI
[56] "FAQ - Flutter." https://flutter.dev/docs/resources/faq. Accessed 14 May. 2019.
[57] "FAQ - Flutter." https://flutter.dev/docs/resources/faq. Accessed 14 May. 2019.
[58] "A Deep Dive Into Transform Widgets in Flutter – Flutter Community ...." 15 Nov. 2018, https://medium.com/flutter-community/a-deep-dive-into-transform-widgets-in-flutter-4dc32cd575a9
[59] Google Developers "Transform (Flutter Widget of the Week)" https://www.youtube.com/watch?v=9z_YNlRlWfA
[60] "Integrating Flutter into an Existing App — Part One: Flutter ... - Medium." 10 Sep. 2018, https://medium.com/@tpolansk/integrating-flutter-into-an-existing-app-part-one-flutter-with-submodules-9b633ff3cf10.
[61] "Hummingbird: Building Flutter for the Web – Flutter – Medium." 4 Dec. 2018, https://medium.com/flutter-io/hummingbird-building-flutter-for-the-web-e687c2a023a8.

sdk. This allows developer to access all the knowledge they need directly from the flutter docs instead of having to research used dependencies. After getting past Flutters initial high learning curve it allows to do a lot with just the basics.

BLoC is very familiar to Flux. However it might be harder to wrap your head around as it employs functional programming. You are not required to enforce any kind of pattern and Flutter also supports the use of patterns like Redux or Mobx.

Overview

One of the biggest things I enjoyed about Flutter is that it shakes all the web dependencies. The move away from javascript allows the use of a statically typed language without the need for any set up. There is also no need for trans compilers like babel to make it backwards compatible. Widgets seem more intuitive way to build UI. Since they work like functions you can treat UI as code. However this break away from web paradigms comes with a steeper learning curve.

What makes Flutter very interesting is its approach to using a 2D engine for UI. By transcending the limitations of the native UI it empowers developers and designers alike to create and share unique, motion and gesture rich UI. In one of their articles about react AirBnb mentioned that they created a unified design language system[62] to make sure their designs were consistent throughout the system. Flutters widget system seems to make it so it is not only easier to do this but also more customizable as you are no longer limited by the Ios or Android UI. However it does come with a cost. Since the UI is no longer Native it can lead to the applications feeling different from actual Native applications as well as update lag since the Flutter engineers will need more time to replicate the feeling of Native UI. It is worth mentioning that flutter is developed by google and they are also in charge of android SDK. This could mean less lag in communication before new updates from the android side.

Performance wise Flutter falls behind native applications. Both React Native and Flutter rely on certain optimization to achieve native-like experience. However from my experience in Flutter the optimizations are easier and more intuitive to implement compared to React Native[63][64]. As there is no need to initialize a javascript runtime Flutter also seems to tackle the initialization time problem that React Native is facing.

Flutter is certainly behind React Native or Apache Cordova when it comes to the Ecosystem, as both of them have already been there for years before Flutter was released. This paired reduced code reusability and steep learning curve can make Flutter seem like an expensive choice. As of currently using Flutter for a large scale project would mean a huge investment into its infrastructure. However

[62] "Building a Visual Language - Airbnb Design." https://airbnb.design/building-a-visual-language/.

[63] "Examining performance differences between Native ... - Thoughtbot." 23 Mar. 2019, https://thoughtbot.com/blog/examining-performance-differences-between-native-flutter-and-react-native-mobile-development.

[64] "What Is Google Flutter? An in Depth Look at the Pros and Cons of This ...." 26 Feb. 2019, https://www.techstuffed.com/what-is-google-flutter-an-in-depth-look-at-the-pros-and-cons-of-this-app-developer/.

despite the still growing community Flutter without a doubt offered the best documentation compared to React Native or Cordova Apache

All in all Flutter is a new kid on the block but it has a lot of potential. Flutter seems to address a lot of issues caused by reliance on web technologies. Although it comes with a cost the developer community seems to be accepting of it as it is seeing huge growth since its release. However it is still too early to see how well it scales. Adopting Flutter for a large scale project at this point is still risky.

Apache cordova

Introduction

Apache Cordova (formerly PhoneGap) is a mobile application development framework owned by Adobe Systems[65]. It enables developers to build applications for multiple platforms using regular web technologies like CSS, HTML, and JavaScript. The resulting applications are achieved with the use of webview technology. They are referred to as hybrid as they bridge native and web applications. They are native applications and have access to certain native APIs, however layout rendering is done via Web views instead of the platform's native UI framework.

Webview

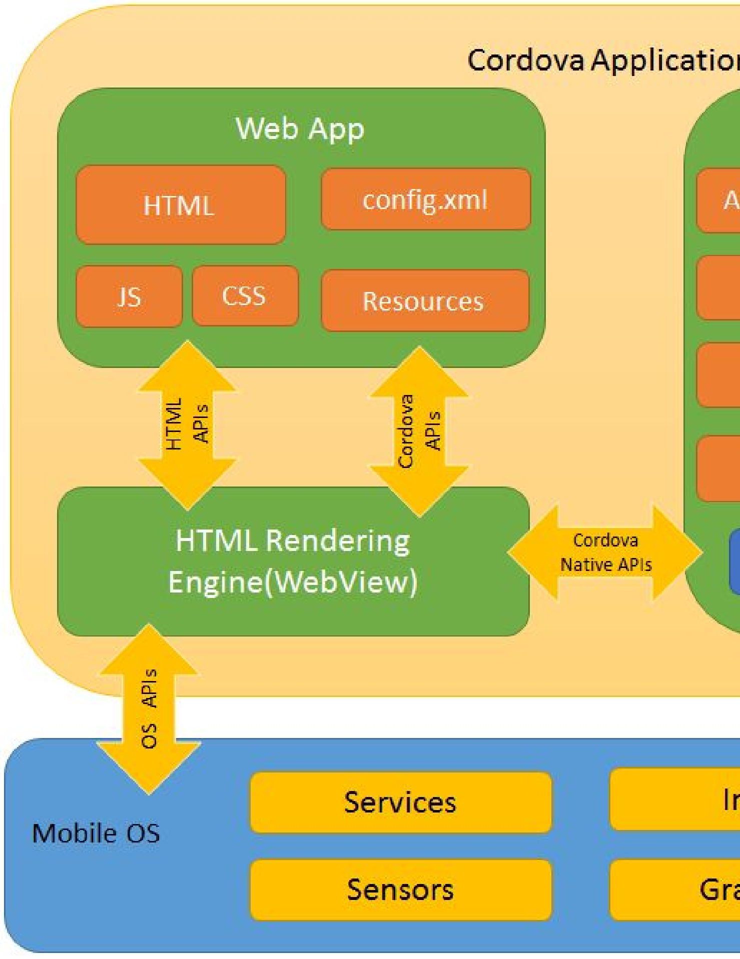
Cordova applications are essentially containers for a webview. WebView is an application component (like a button or a tab bar) that is used to display web content within a native application. You can think of a WebView as a web browser without any of the standard user interface elements, such as a URL field or status bar. Cordova takes the web application you developed, runs a local server and hosts it and displays it through said webview.

The web application running inside this container is just like any other web application that would run within a mobile browser—it can open additional HTML pages, execute JavaScript code, play media files, and communicate with remote servers.

[65] "PhoneGap, Cordova, and what's in a name?." 19 Mar. 2012, https://phonegap.com/blog/2012/03/19/phonegap-cordova-and-whate28099s-in-a-name/.

The Native containers change according to the OS but internally the web pages remain the same. It accesses the native APIs through plugins developed in native code.[66] [67]

The following diagram shows a high-level view of the Cordova application architecture:



[66] "Ionic Article: What is Apache Cordova?." https://ionicframework.com/enterprise/resources/articles/what-is-apache-cordova.
[67] "What is Cordova and how does it work ? | SAP Blogs." 27 Jul. 2014, https://blogs.sap.com/2014/07/27/what-is-cordova-and-how-does-it-work/

Credit: Official Apache Cordova documentation[68]

The web application is hosted in the webview with which it communicates using an API provided by Cordova. Since The Webview is a native element it can effectively communicate with the OS. Plugins are used to translate needed native functionality so it can be used by the web application.

Plugins

Cordova comes with a set of core plugins which allow access to the device's accelerometer, camera, compass, file system, microphone, and more[69].They can also be extended allowing developers to add more functionalities that can be directly accessed from JavaScript. They are essential part of Cordova as they act as a bridge between your web application and the native layer.

Web technologies

Cordova applications are built using standard web technologies JS, HTML and CSS. Since Cordova applications are basically web applications it is possible to use any web technology stack.

Frontend UI frameworks

Cordova by itself does not offer any frontend framework to work with. However there are multiple tools and frameworks built on top of Cordova which provide the functionality which allows to build native-like applications. Most notable ones being Ionic, Jquery Mobile, Onsen UI and Framework7.

Implementation

Source code
https://github.com/Louvki/ionic-todo

Set up

Cordova offers using either Cross-platform (CLI) workflow or Platform-centered workflow. Platform-centered workflow allows more low level control and native access for the specific platform you are targeting. However it makes it harder to achieve cross platform development because the lack of a higher-level tool means separate build cycles and plugin modifications for each platform. For this reason the Cross-platform (CLI) workflow is better when working on building a cross platform application.

However I will not be using Cordova directly as it requires a lot of overhead to replicate the feeling or look of a native application. Instead I will be using the UI framework Ionic which is based on Cordova. Although Ionic can be platform agnostic it also has official integration with Angular so I will be using that.

Install Node.js
Install ionic using the npm package manager
Install android studio
Set up the path for android sdk

I used the ionic command line to initialize a blank application.

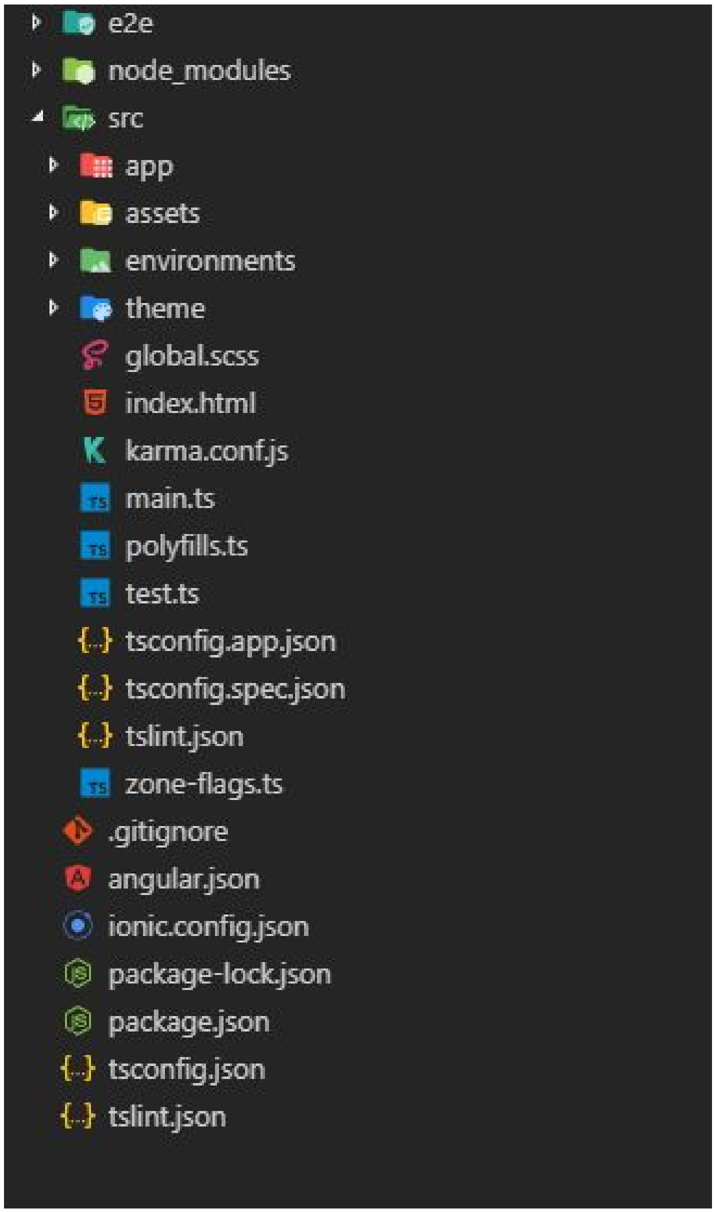
```
ionic start myApp blank
```

[68] https://cordova.apache.org/docs/en/latest/guide/overview/index.html

[69] "Cordova support by platform - Apache Cordova." https://cordova.apache.org/docs/en/latest/guide/support/

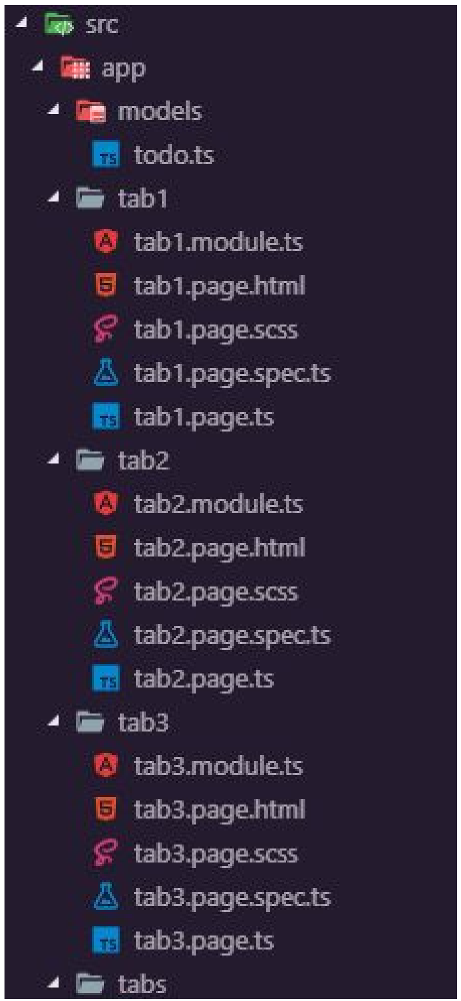
Since Ionic is basically a web application I can just call `ionic serve` and it will host my app on localhost port:8100. This way I can use chrome developer tools to debug my application. I can also test the application on android or ios devices by calling the command `ionic cordova build android` or `ionic cordova build ios`

File structure



```
e2e
node_modules
src
  app
  assets
  environments
  theme
  global.scss
  index.html
  karma.conf.js
  main.ts
  polyfills.ts
  test.ts
  tsconfig.app.json
  tsconfig.spec.json
  tslint.json
  zone-flags.ts
.gitignore
angular.json
ionic.config.json
package-lock.json
package.json
tsconfig.json
tslint.json
```

e2e - files for end-to-end testing using the protractor library. Protractor is a framework written by the Angular team and is used with test runners like Karma.
node_modules - external modules installed with the npm package manager
src/app - all your application related code
assets - assets like images, audio files etc.
environments - for switching between production and development environments
theme - style variables describing your apps theme
global.scss - global css files used by ionic
index.html - entry point to your web app
karma.conf.js - Karma is a unit testing framework. All .spec files are used by karma. Karma also requires the **tests.ts** file
main.ts - this is where angular bootstraps your application
polyfill.ts - A polyfill is a bit of code that adds functionality to the browser and normalizes browser differences. This file is configuring polyfills for your application.
tsconfig - app wide typescript configuration files
tslint - app wide linting options for typescript.
angular.json - angular configuration file
Package.json
package.json - dependency and other metadata configuration file.
tsconfig - project wide typescript configuration files
tslint - project widelinting options for typescript.



```
src
  app
    models
      todo.ts
    tab1
      tab1.module.ts
      tab1.page.html
      tab1.page.scss
      tab1.page.spec.ts
      tab1.page.ts
    tab2
      tab2.module.ts
      tab2.page.html
      tab2.page.scss
      tab2.page.spec.ts
      tab2.page.ts
    tab3
      tab3.module.ts
      tab3.page.html
      tab3.page.scss
      tab3.page.spec.ts
      tab3.page.ts
    tabs
```

Implementation
I am using Typescript and Angular for my implementation. I am using rxjs library for data management. This library allows the use of tools like Observable streams and Behaviour subject which allows the handling of data in a more reactive manner.

Development Speed

Live Reload[70] is a tool that automatically reloads the browser when it detects changes. It watches a specific path and when a file or directory is updated, it will trigger a full browser reload. In cases where CSS is changed, live reload will inject the changed CSS instead of reloading the entire page. This is the equivalent of a Hot reload in Flutter or React Native applications. Downside of it is it needs to be set up manually if Apache Cordova is not used in tandem with an UI framework.

Browser debugging. All modern browsers have an in inbuilt debugger. This makes them not only easy to access and set-up but they are also very powerful. For example chrome developer tools offers device emulation(makes the webpage act like as if it would be on a phone), direct element editing, network request monitoring, performance recording and many other tools.[71][72][73]

Established component libraries. UI frameworks like Ionic or Framework7 offer extensive libraries of premade components which work great for tailoring a mobile experience.

Large community. Cordova Apache has been released back in 2011. There are multiple popular frameworks built on this technology.

Purely Web. Full freedom to use any web based tool or library within your project.

Strengths and weaknesses

Since Apache Cordova applications are essentially web applications they come with the overhead of having to replicate native UI and gestures[74]. To enable support for more native like gestures you are required to use polyfills (a polyfill is a bit of code that adds functionality to the browser and normalizes browser differences) or libraries[75]. UI frameworks do negate the problem to some extent by providing pre styled components and polyfills which are meant to imitate native behaviour. However these frameworks do come at the cost of flexibility as they enforce the use of a specific technology stack.

In the end whether you are using an UI framework or are choosing to build purely your own application it can result in managing more dependencies[76] than you initially anticipated. This

[70] "Live Reload All the Things: Ionic CLI's Latest Features | The Ionic Blog." 3 Sep. 2014, https://blog.ionicframework.com/live-reload-all-things-ionic-cli/
[71] "Chrome DevTools - Google Developers." https://developers.google.com/web/tools/chrome-devtools/.
[72] "Listing Top Browser Debugging Tools For Developers | 61DesignStreet." 9 Mar. 2016, http://www.61designstreet.com/blog/top-browser-debugging-tools/
[73] "What are browser developer tools? - Learn web development ... - Mozilla." 18 Mar. 2019, https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_are_browser_developer_tools.
[74] "Next Steps - Apache Cordova - The Apache Software Foundation!." https://cordova.apache.org/docs/en/latest/guide/next/.
[75] "How to implement long press gestures in Ionic 4 – madewithply ...." 9 Oct. 2018, https://medium.com/madewithply/ionic-4-long-press-gestures-96cf1e44098b.
[76] "Apache Cordova: after 10 months, I won't using it ... - Hacker News." 6 Jul. 2015, https://news.ycombinator.com/item?id=9838169.

also can be a cause of problems as some plugins have compatibility issues with different devices and platforms. Another thing to consider is the update lag. Cordova applications have to not only replicate the UI but also gestures and animations.

Of course using web technologies does come with its upsides as well. If working purely with Cordova Apache there are no restraints on what web based technology or library can be used within the project.

When it comes to performance webview applications simply do not perform as well as native ones.[77] This is due to them being limited by the power of the rendering engine[78] on the platform that you are supporting. Since the browser acts like a middle man you will not have the connection to the full hardware performance or the entire memory stack. This means you will need to optimize memory management when working with GPU intensive tasks (loading images, animations). Even with optimizing Issues may arise when working with an application that handles large amounts of data.[79]

Learning curve

JS, HTML, CSS. Cordova apache is fully web based. From debugging to developing it will not differ much from making a regular web application. You can use all the tools you are familiar from web development.

The only real learning curve you will encounter is if you are using UI frameworks or plugins. Most of them time you will not need to write platform specific code but if you do Cordova offers some simple APIs for checking which platform you are on.

Overview

Some well made hybrid applications are hard to distinguish from native ones. However recreating the feeling of a native application on the browser can be daunting task. It is easier with the use of UI frameworks or external libraries however that also means introducing dependencies.

Another huge drawback is that Cordova Apache is its reliance on webview technology. The performance and speed of Cordova applications directly depends on the browser engine used. Since Cordova is so far detached from native it suffers from update lag the most. Certain browser engines might not support functionality you require thus further prolonging it.

[77] "What is Cordova Framework? – Mindfire Solutions – Medium." 5 Feb. 2018, https://medium.com/@mindfiresolutions.usa/what-is-cordova-framework-be59fb6124f8.
[78] "Change default webkit on Apache Cordova - Android - Stack Overflow." https://stackoverflow.com/questions/18805611/change-default-webkit-on-apache-cordova-android.
[79] "What's Your View on Apache Cordova? - Mutual Mobile." 23 Jul. 2014, https://mutualmobile.com/resources/whats-view-apache-cordova.

Cordova Apache has a huge community surrounding it. However it is split up between different adoptions of it. What Flutter and React Native have for them is that they have their own ecosystem. Meanwhile Cordova is mostly used in tandem with UI frameworks whose communities although working on the same underlying technology nevertheless split up.

Cordova has its own niche spot that it fills pretty well. Although it might not be great for building an intricate large scale applications, webview technology shine when creating smaller applications due to development cost. Compared to React Native or Flutter, Cordova without a doubt offers the lowest barrier for entry. It uses purely web technologies which makes hiring developers cheaper and it enables the most code-reuse. Webviews can also be integrated within an existing native application.[80] However Cordova has been slowly falling out of favor with the developer community. In the 2019 survey 7.1% responded to using Cordova. It has been a drop from last years 8.5%. This is most likely due to the rise of progressive web applications as they offer the same functionality without the need for all the dependencies.

[80] "Embedding WebViews - Apache Cordova." https://cordova.apache.org/docs/en/latest/guide/hybrid/webviews/.

Conclusion

Based on the theory that cross platform frameworks try to have the best of the web and native development and by analysing strengths and weaknesses of said paradigms I have outlined the goals cross platform frameworks try to achieve: **Code reuse, Performance, Flexibility and Low cost.** Based on the goals and general information I outlined the likely problems of cross platform development to be **underperformance, inflexibility, dependencies and low adoption.**

After my research into the current available cross platform frameworks I have chosen **React Native, Flutter** and **Cordova Apache** as three most promising frameworks. I have analyzed each framework on the previously outlined goals and problems. Following is the summary of findings and comparison of the frameworks.

Code reuse
Cordova without a doubt offers the most code reuse between all platforms. As it is web based any device capable of opening a browser will be able to run web applications. React is the runner-up as most of its UI is written in a weblike manner. Flutter offers the least code reuse as it currently does not support the web. Any of the frameworks code reuse can be improved with the use of software state management patterns.

Performance
Cross platform applications (even the ones who run natively) still use more system resources than regular native applications. However unless the goal is to make something extremely CPU intensive no problems should arise. With proper optimization and implementation native-like performance can be achieved. The performance of course also depends on the scale of the project and technology stack used.

Frameworks like Flutter, React Native, Nativescript will be more performant compared to Cordova purely due to the fact that the code is being run natively. It seems like Flutter is the top contender for performance as it managed to reduce initialization times, something javascript bridge based frameworks (Nativescript, React Native) struggled with.

Flexibility
It is unavoidable that as an application grows there will be a need to implement some android or ios specific code. However if a framework often relies on this platform specific code and does not offer proper abstractions to native APIs instead of developing one codebase you might end up with three. This was one of the largest problems for AirBnB when working with React Native as it fragmented their team and workflow since they had to maintain Android, ios and React

Native code.[81] Cordova does not suffer from these problems as it is web based however this also comes with a downside of having to create bridges (plugins) to access native functionality and the overhead of replicating it.

Flutter is canvas based so it avoids the problem by not having to rely on native UI and it offers inbuilt tools for creating it. This also seems to open doors for more motion rich and unique design choices as it no longer relies on the underlying gesture system. Which was outlined as one of the problems for AirBnb when working with React Native. When it comes to gestures Cordova falls short as well as it does not have access to native gestures.

Dependencies and learning curve

When it comes with dealing with external dependencies Flutter is the clear winner. By not basing itself on web technologies it creates a fully isolated ecosystem. However this smoother and less bug prone development flow does come with a much steeper learning curve.

React Native relies on web technologies and this allows for a lower learning curve than Flutter. Cordova Apache struggles with dependencies due to its community managed plugins as well as due to limitations of the browser engine. Relying so heavily on the browser engine can lead to update lags. Meanwhile as all of the plugins are community maintained there is no guarantee they will be up to date.

Adoption

React Native is the clear winner due its large and growing community, facebook backing and due to implementing already established libraries like React. However Flutter has entered the cross platform race strongly and is quickly gathering community support.

Cordova Apache is the most mature and offers a wide variety of mature frameworks based on it. However based on Stack overflow it is decreasing in popularity. In the 2019 survey 7.1% responded to using Cordova. It has been a drop from last years 8.5%. It is also one of the most dreaded technologies as over 63.3% of the developers who know *Cordova* reported not enjoying working with it.

Cost

Using any cross platform framework will be cheaper than native development however no matter the choice all of them will be more expensive than web. All frameworks come with their own learning curve as well as the learning curve for the technology stack to be used with them. Frameworks which employ web technologies (React Native, Nativescript, Cordova) will generally be cheaper to implement. Cordova being practically a web application is naturally the cheapest option. Flutter straying furthest away from web technologies and paradigms is currently the most expensive.

[81] "Building a Cross-Platform Mobile Team – Airbnb Engineering & Data ...." 19 Jun. 2018, https://medium.com/airbnb-engineering/building-a-cross-platform-mobile-team-3e1837b40a88.

Reference list

Mobile operating system market share worldwide. Accessed May. 2019.
http://gs.statcounter.com/os-market-share/mobile/worldwide.

"Dependency hell - Wikipedia". Accessed May. 2019.
https://en.wikipedia.org/wiki/Dependency_hell.

Stack Overflow Developer Survey 2019 - Stack Overflow Insights."
https://insights.stackoverflow.com/survey/2019.

Stack Overflow Developer Survey 2018 - Stack Overflow Insights."
https://insights.stackoverflow.com/survey/2018

React Native Github. Accessed May. 2019.
https://github.com/facebook/react-native

React native app showcase. Accessed May. 2019.
https://facebook.github.io/react-native/showcase.

Flutter github. Accessed May. 2019.
https://github.com/flutter/flutter. Accessed May. 2019.

Flutter app showcase. Accessed May. 2019.
https://flutter.dev/showcase

Native flutter growth comparison to react native. Accessed May. 2019.
https://star-history.t9t.io/#facebook/react-native&facebook/react-native&flutter/flutter.

Ionic github. Accessed May. 2019.
https://github.com/ionic-team/ionic

Framework 7 github. Accessed May. 2019.
https://github.com/framework7io/Framework7

Jquery mobile github. Accessed May. 2019
https://github.com/jquery/jquery-mobile/

Nativescript github. Accessed May. 2019.
https://github.com/NativeScript/NativeScript

Nativescript growth comparison to react native. Accessed May. 2019.
https://star-history.t9t.io/#NativeScript/NativeScript&NativeScript/NativeScript&facebook/react-native

Xamarin Forms github. Accessed May. 2019.
https://github.com/xamarin/Xamarin.Forms

Official Xamarin documentation

https://docs.microsoft.com/en-us/xamarin

React Native: Bringing modern web techniques to mobile - Facebook ...." 26 Mar. 2015, https://code.fb.com/android/react-native-bringing-modern-web-techniques-to-mobile/

"React Native App how many threads? Execution Model ... - WebKJ." https://www.webkj.com/react-native/react-native-app-how-many-threads.

"How React Native Works ? | Codementor." 16 Aug. 2018, https://www.codementor.io/saketkumar95/how-react-native-works-mhjo4k6f3.

"React Native: What it is and how it works – We Talk IT – Medium." 12 Jun. 2017, https://medium.com/we-talk-it/react-native-what-it-is-and-how-it-works-e2182d008f5e

React wikipedia page
https://en.wikipedia.org/wiki/React_(JavaScript_library)

Top starred repositories. Accessed May. 2019.
https://github.com/search?p=1&q=stars%3A%3E1&ref=searchresults&type=Repositories&utf8=%E2%9C%93

Flux state management pattern in depth overview
https://facebook.github.io/flux/docs/in-depth-overview.html

Using Postal.js as Flux Dispatcher in your modular React application ...." 24 Jun. 2015, https://towardsdatascience.com/using-postal-js-as-flux-dispatcher-in-your-modular-react-application-with-turris-js-1355e9feacc9.

"Introducing Hot Reloading · React Native - Facebook Open Source." 24 Mar. 2016, https://facebook.github.io/react-native/blog/2016/03/24/introducing-hot-reloading.

"Introducing JSX – React."
https://reactjs.org/docs/introducing-jsx.html.

"React Native at Airbnb: The Technology – Airbnb Engineering - Medium." 19 Jun. 2018, https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838

"JSX - a faster, safer, easier JavaScript."
https://jsx.github.io/.

"Performance Limitations of React Native and How to Overcome Them." 19 Jun. 2016, https://medium.com/@talkol/performance-limitations-of-react-native-and-how-to-overcome-them-947630d7f440

"6 Tips You Want to Know About React Native Performance — Simply ...." 7 Jun. 2017, https://www.simplytechnologies.net/blog/2017/6/6/6-tips-you-want-to-know-about-react-native-performance

React Native at Airbnb: The Technology – Airbnb Engineering - Medium." 19 Jun. 2018, https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838.
React Native at Airbnb: The Technology – Airbnb Engineering - Medium." 19 Jun. 2018, https://medium.com/airbnb-engineering/react-native-at-airbnb-the-technology-dafd0b43838.

"Would Airbnb Have Fared Better With NativeScript Instead of React ...." 27 Jun. 2018, https://www.nativescript.org/blog/would-airbnb-have-fared-better-with-nativescript-instead-of-react-native

"State of React Native 2018 - Facebook Open Source." 14 Jun. 2018, https://facebook.github.io/react-native/blog/2018/06/14/state-of-react-native-2018

"Pros and cons of React Native - mobile apps with JavaScript · Devbridge." 28 Apr. 2016, https://www.devbridge.com/articles/pros-cons-of-react-native-crash-course/.

Flutter's first official release.
https://github.com/flutter/flutter/releases/tag/v0.0.6

"Language Tour | Dart."
https://dart.dev/guides/language/language-tour

"FAQ - Flutter."
https://flutter.dev/docs/resources/faq

"Skia Graphics Library." https://skia.org/.

"List of state management approaches - Flutter."
https://flutter.dev/docs/development/data-and-backend/state-mgmt/options..

"BLoC design pattern with Angular – lacolaco-blog – Medium." 23 May. 2018, https://medium.com/lacolaco-blog/bloc-design-pattern-with-angular-1c2f0339f6a3.

"Architect your Flutter project using BLOC pattern – FlutterPub – Medium." 26 Aug. 2018, https://medium.com/flutterpub/architecting-your-flutter-project-bd04e144a8f1.

"Flutter - Reactive Programming - Streams - BLoC - Didier Boelens." 20 Aug. 2018, https://www.didierboelens.com/2018/08/reactive-programming---streams---bloc/.

"Hot reload - Flutter."
https://flutter.dev/docs/development/tools/hot-reload

"What is Reactive Programming? – Keval Patel – Medium." 12 Dec. 2016, https://medium.com/@kevalpatel2106/what-is-reactive-programming-da37c1611382.

"Architect your Flutter project using BLOC pattern – FlutterPub – Medium." 26 Aug. 2018, https://medium.com/flutterpub/architecting-your-flutter-project-bd04e144a8f1.

"Flutter Theme Class | Flutter By Example."
https://flutterbyexample.com/flutter-theme-class/

"Flutter performance profiling - Flutter."
https://flutter.dev/docs/testing/ui-performance.

"How fast is Flutter? I built a stopwatch app to find out. - freeCodeCamp ...." 18 Mar. 2018, https://medium.freecodecamp.org/how-fast-is-flutter-i-built-a-stopwatch-app-to-find-out-9956fa0e40bd.
"A Deep Dive Into Transform Widgets in Flutter – Flutter Community ...." 15 Nov. 2018, https://medium.com/flutter-community/a-deep-dive-into-transform-widgets-in-flutter-4dc32cd575a9

Google Developers “Transform (Flutter Widget of the Week)“
https://www.youtube.com/watch?v=9z_YNlRlWfA

"Integrating Flutter into an Existing App — Part One: Flutter ... - Medium." 10 Sep. 2018, https://medium.com/@tpolansk/integrating-flutter-into-an-existing-app-part-one-flutter-with-submodules-9b633ff3cf10.

"Hummingbird: Building Flutter for the Web – Flutter – Medium." 4 Dec. 2018, https://medium.com/flutter-io/hummingbird-building-flutter-for-the-web-e687c2a023a8.

"Building a Visual Language - Airbnb Design."
https://airbnb.design/building-a-visual-language/.

"Examining performance differences between Native ... - Thoughtbot." 23 Mar. 2019, https://thoughtbot.com/blog/examining-performance-differences-between-native-flutter-and-react-native-mobile-development.

"What Is Google Flutter? An in Depth Look at the Pros and Cons of This ...." 26 Feb. 2019, https://www.techstuffed.com/what-is-google-flutter-an-in-depth-look-at-the-pros-and-cons-of-this-app-developer/.

PhoneGap, Cordova, and what's in a name?." 19 Mar. 2012,
https://phonegap.com/blog/2012/03/19/phonegap-cordova-and-whate28099s-in-a-name/.

"Ionic Article: What is Apache Cordova?."
https://ionicframework.com/enterprise/resources/articles/what-is-apache-cordova.

"What is Cordova and how does it work ? | SAP Blogs." 27 Jul. 2014,
https://blogs.sap.com/2014/07/27/what-is-cordova-and-how-does-it-work/

Official Apache Cordova documentation overview
https://cordova.apache.org/docs/en/latest/guide/overview/index.html

Cordova support by platform - Apache Cordova."
https://cordova.apache.org/docs/en/latest/guide/support/

"Live Reload All the Things: Ionic CLI's Latest Features | The Ionic Blog." 3 Sep. 2014,
https://blog.ionicframework.com/live-reload-all-things-ionic-cli/

"Chrome DevTools - Google Developers."
https://developers.google.com/web/tools/chrome-devtools/.

"Listing Top Browser Debugging Tools For Developers | 61DesignStreet." 9 Mar. 2016,
http://www.61designstreet.com/blog/top-browser-debugging-tools/

"What are browser developer tools? - Learn web development ... - Mozilla." 18 Mar. 2019,
https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_are_browser_developer_tools.

"Next Steps - Apache Cordova - The Apache Software Foundation!."
https://cordova.apache.org/docs/en/latest/guide/next/.

"How to implement long press gestures in Ionic 4 – madewithply ...." 9 Oct. 2018, https://medium.com/madewithply/ionic-4-long-press-gestures-96cf1e44098b.

"Apache Cordova: after 10 months, I won't using it ... - Hacker News." 6 Jul. 2015, https://news.ycombinator.com/item?id=9838169.

"What is Cordova Framework? – Mindfire Solutions – Medium." 5 Feb. 2018, https://medium.com/@mindfiresolutions.usa/what-is-cordova-framework-be59fb6124f8.

"Change default webkit on Apache Cordova - Android - Stack Overflow." https://stackoverflow.com/questions/18805611/change-default-webkit-on-apache-cordova-android.

"What's Your View on Apache Cordova? - Mutual Mobile." 23 Jul. 2014, https://mutualmobile.com/resources/whats-view-apache-cordova.

"Embedding WebViews - Apache Cordova." https://cordova.apache.org/docs/en/latest/guide/hybrid/webviews/.