Research hybrid development frameworks

Marco Ketelaars en Rick van Wijk

Introduction

In this research paper we take a look at some of the many hybrid frameworks out there and compare them to each other. At the end of this document, a conclusion will be drawn about which of the hybrid frameworks is the best or, if there's no clear winner, which ones excel in specific use cases.

The research will be done from the perspective of a start-up company that wants to build apps for multiple platforms.

Main research question

What hybrid framework is the best/most suitable hybrid framework?

Sub questions

What is a Hybrid framework?

Which Criteria should we prioritize?

What hardware do we need?

What platforms are we going to develop for?

How do the frameworks compare based on our criteria?

What are the advantages of the frameworks that fulfill our criteria?

Motivation

By doing research on hybrid frameworks, we'll know what hybrid framework will be best/most suitable to use for the hybrid apps we're going to develop later in the semester.

Project context

The project is a livestream app users can use to stream concerts. Other users can then watch the concerts online. The app needs to consider whether the organizers of the app gave permission to livestream.

Plan of action

Five hybrid frameworks will be investigated by doing library/desktop and workshop research. For each framework a POC will be executed in order to see how the framework works and based on the results we will summarize what the strengths and weaknesses are.

Framework list

List of frameworks that are going to be researched, we chose some of the more popular frameworks as a first step.

- https://flutter.dev/
- https://reactnative.dev/
- https://ionicframework.com
- https://dotnet.microsoft.com/apps/xamarin
- https://nativescript.org

Criteria

- Ease of use
- Maintainability
- Speed
- Community
- Documentation
- OS support
- Scalability

Research questions

What is a hybrid framework?

Hybrid apps are built from a single codebase and are web-based but developed in a way that they can also be used natively. Hybrid frameworks are the technologies with which you can develop the hybrid apps. Most hybrid frameworks use web technologies like HTML and CSS to create the front-end UI, by embedding in the App using a browser, it can use plugins to communicate with underlying systems and use the same functionality as a native app. Others like Xamarin build target

Which criteria should we prioritize?

OS support

OS support is not that important for us, all tested frameworks support enough versions of Android and iOS that most users can make use of our apps. We have no need to support systems beyond that.

Scalability

The apps we'll create are small so scalability while useful when keeping an eye on the future is not currently very important.

Ease of use

Getting started as soon as possible is always a good thing. If a framework is easy to use it will give us more time to develop. As ca start-up we should get our first app out quickly.

Maintainability

It's important that the apps we create are easy to maintain, this needs to be at least a score of 7.

Speed

Speed is less important than maintainability but if it's below a score of 5, it might impact UX too much.

Documentation

It's important that the documentation is well written, easy to understand and that it’s easy to find information in.

Community

The size of the community matters. Especially when documentation might be lacking in some respects. A bigger community means more information and examples is available outside of official sources. Community is also important when debugging some things by yourself doesn’t work out.

What hardware do we need/does the framework need access to?

The concept requires the followering

- GPS location
- Camera
- Microphone

What platforms are we going to develop it for?

- Android
- IOS

How do the frameworks compare based on our criteria?

We will judge each framework by giving a score of 1 to 10 based on our own findings while making Proof of Concepts, reviews and our own judgement from what we can find about the frameworks on the internet.

The proof of concepts will be simple applications, for example the Proof of Concept made for the Xamarin application was a simple counter App.

Framework	Ease of use	Maintainability	Speed	Community	Documentation	OS support	Scalability
Flutter	7	8	9	8	9	8	8
React Native	9	4	4	8	9	7	6
Ionic	8	8	7	7	8	7	7
Xamarin	7	8	4	6	6	7	8
NativeScript	8	8	7	5	8	8	8

Below we will describe findings that contextualize the above score where necessary.

Xamarin

Because it's supported by Visual Studio, the project itself was easy to set up. There does seem to be a bit of a learning curve when it comes to programming it the way you're supposed to. The XAML of Xamarin forms should display data via data bindings, the documentation for data bindings was lacking in terms of binding to properties. Xamarin is the only of the researched frameworks where the recommended IDE (visual studio) costs money to use. Free versions of Visual Studio have limitations on commercial usage. It supports Android, iOS and Windows Phone.

Ionic

Through NPM and Node it was easy to install, there were issues with deprecated packages depending on which style and template was chosen. For the proof of concept several combinations were tested, there were some issues because a list of commands was necessary to make the project deployable on mobile devices. Every issue however could swiftly be resolved using documentation and community answers. Because of the web-based nature of Ionic you can test the application in your browser, of course the functionality that depends on being a mobile App are unavailable at that time. Android and iOS are supported.

React-Native

The set-up required Node, JDK and Android Studio. This does take a while to set up, but since you need those resources for a lot of other types of development environments as well it's worth your time. The community is big. A lot of components/modules come from the community. Unfortunately, this makes

React-Native not the best framework when It comes down to maintainability. Debugging with React-Native doesn't have many features, but it does allow you to use the Chrome Devtools with a VScode extension and some extra dependencies. The live app preview that updates on each safe (autosave is optional) makes developing the frontend of the app easier. Another good thing about React-Native is that it has Expo which helps you develop, build, deploy and iterate for all the different platforms.

Flutter

The set-up was comparable to React-Native. Both primarily recommend using the Android emulator, so it was no surprise the set-ups were very similar. You can have a nice workflow with VScode by installing the Flutter extension. There's a lot of handy debugging features, but if you can't solve a problem yourself you can always ask the community or go to the dev documentation. Viewing/debugging on different devices/browsers is no problem. You can even view the app as a windows desktop application with the Flutter extension in VScode. The 'Hot Reload' option (on save the app gets updated) only works for the simple things like changing styles of components/widgets so if you want to add variables or change the root widget you will have to restart the debugging session which is a minor inconvenience of Flutter.

NativeScript

Just like React-Native and Flutter the set-up required Node, JDK and Android Studio (plus the NativeScript CLI). However, I encountered gradle problems (which I didn't have when using React-Native and Flutter) and when I tried to search for the solution online, I found out that the community was not as big as React-Native and Flutter. I tried debugging it myself, but nothing seemed to work. I did find out that NativeScript allows you to choose from a wide variety of JavaScript frameworks you want to build your app with (e.g. Vue, Angular, React, Svelte, etc.). This makes it a very flexible hybrid framework. It also has plenty of documentation on how to debug and setup your workflow. The community doesn't seem to be that big, but that's probably because the users of NativeScript are split up between the JavaScript frameworks you're allowed to choose.

Flutter/Ionic/NativeScript - which one is more suitable to the concept?

The above frameworks fullfilled our criteria. Now we need to research which one is the most suitable for our concept.

Concept

Visitors to a pop concert can (if allowed by the organisation of the concert) stream live video (and music) to a server with this app. Users can then log on to that server and follow the concert live on screen.

Flutter

- Development for Android and iOS
- Meets the hardware requirements; it supports the hardware
- Has the option to use the third-party SDK Agora for livestreaming and a Youtube player plugin suitable for livestreaming

Ionic

- Ionic can livestream via a plugin from Bambuser.

- Uses web technology

NativeScript

- Development for Android and iOS
- Meets the hardware requirements; it supports the hardware
- Doesn't have a plugin/SDK for livestreaming

Conclusions & Recommendations

The result of the research is that Flutter is the most suitable hybrid framework for our concept. Because besides passing the general criteria it also has plenty of plugins and options for livestreaming.

Ionic was a close second place that could be used if Flutter turns out to be less suitable during development.

NativeScript misses livestreaming capabilities that are required for our project.

Xamarin and React-native don't meet the criteria defined as part of the research. Both of them fail on speed & performance. React-native also lacks maintainability.

Sources

https://ionic.io/resources/articles/what-is-hybrid-app-development

https://www.icharts.net/the-pros-and-cons-of-xamarin-review/

Flutter vs Native vs React-Native: Examining performance - https://medium.com/swlh/flutter-vs-native-vs-react-native-examining-performance-31338f081980

React-Native review - https://engineering.velocityapp.com/scalable-react-native-mobile-development-cfab6c3c7499

Flutter vs React-Native community size - https://dev.to/mjablecnik/how-large-is-flutter-vs-react-native-community-in-2021-2df5

A shared experience - https://blog.codemagic.io/flutter-vs-ios-android-reactnative-xamarin/

https://medium.com/asos-techblog/flutter-vs-react-native-for-ios-android-app-development-c41b4e038db9

https://www.mobileappdaily.com/best-hybrid-app-frameworks

https://medium.com/swlh/flutter-vs-native-vs-react-native-examining-performance-31338f081980

https://visualstudio.microsoft.com/vs/compare/

https://docs.microsoft.com/en-gb/xamarin/xamarin-forms/xaml/xaml-basics/get-started-with-xaml?tabs=windows

https://www.altexsoft.com/blog/engineering/performance-comparison-xamarin-forms-xamarin-ios-xamarin-android-vs-android-and-ios-native-applications/