IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Comparison of Flutter with Other Development Platforms

[1] Achal Agrawal, [2]Amit Agrawal, [3]Rahul Arya, [4]Hardik Jain, [5]Jyoti Manoorkar,

[1]Student, [2] Student, [3] Student, [4] Student, [5] Assistant Professor

[1,2,3,4,5]Department of Computer Science and Engineering, Dr. Vishwanath Karad MIT WPU, Pune, India

Abstract: This study has been undertaken to indicate the benefits of using flutter over other app development platforms. It consists of working of different platforms and their role in application development. In today's world ease of development is the thing all developers are looking for and flutter has come up with the required platform to support development of applications for both Android and iOS. Developers are enforced to either construct the same application numerous times for various OS (operating systems) or accept a low-quality similar solution that trades native speed and accuracy for portability. Flutter is an open-source SDK for developing high-performance and more reliable mobile applications for operating systems like iOS and Android.

Keywords: Flutter, iOS, Android, Cross-platform Developement

1. Introduction

To dispense it to most of the users, a mobile application needs to familiarize itself with two independent platforms which are Android and iOS. These two platforms share immense dissimilarities which often necessitate different skill sets for developing. For example Java or Kotlin for Android and Object-C or Swift for iOS. Hence, developers and companies usually struggle to deal with the complex nature involved in developing cross-platform applications.

Cross-platform frameworks that show resemblance to Flutter are discussed and implemented by various companies numerous times formerly. Still, neither of them suffices to satisfy the requirement of industrial development. In spite of the ineffective precursors, Flutter which is backed up by Google, draw attention and developers find it easier to use as well. Flutter application can also run equally on both platforms, consequently decreasing the cost and complexity of application creation across iOS and Android. Flutter is fully built from scratch and around August 2017 only Google used it for commercial projects.

2. Ease of Use

Flutter is a cross-platform framework that targets developing high-performance mobile applications. Besides running on Android and iOS flutter applications also run on Fuschia. Flutter is exceptional because it is dependent on the device's OEM widgets rather than consuming web views. Flutter uses a high-performance rendering engine to render each view component using its own. This provides a chance to build applications that are as high-performance as native applications can be.

3. FLUTTER ARCHITECTURE

In view of architecture, the engine's C or C++ code involves compilation with Android's NDK and LLVM for iOS respectively, and during the compilation process, the Dart code is compiled into native code. Hot reload feature in Flutter is called as Stateful hot reload and it is a major factor for boosting the development cycle. Flutter supports it during development. Stateful hot reload is implemented by sending the updated source code into the running Dart Virtual Machine (Dart VM) without changing the inner structure of the application, therefore the transitions and actions of the application will be well-preserved after hot reloading.

3.1 Dart

In Flutter, every application is written with the help of Dart. Google has developed and maintained a programming language called Dart. It is extensively used inside Google and it has been verified to have the proficiency to develop enormous web applications, such as AdWords. Flutter application renews the view tree on every new frame even when few other systems use reactive views. This behavior leads to a drawback that many objects, which might survive for a singular frame, will be created. As Dart is a modern programming language, it is optimized to handle this scenario in memory level with the help of "Generational Garbage Collection".

3.2 Flutter/Dart UI management

Flutter uses widgets as the main concept within in the code. Widget is the nickname for every component part that is built in Flutter. This could mean a box or a text that is referred to as a widget. A noticeable part of the widgets is that they are created by the Flutter developers to look native and developers are able to fully customize these to their liking.

3.3 Flutter compiling

The way Flutter works when running on Android is by compiling the developers written Dart code to native with the help of AOT compiling. These together with x86 libraries that were created when compiling the dart code, are put into a runner and built as an APK. Flutter runs similar on iOS system, but instead the Dart code compiles into ARM library and is later put as a runner and built as an .ipa.

4. COMPARISON OF FLUTTER WITH OTHER DEVLOPMENT PLATFORMS

4.1 Native Mobile Applications

The meaning of native in the field of mobile applications refers to applications that are built to run on a specific platform or OS. There are numerous languages that can be used for building native mobile applications and a few of examples of these are: Kotlin, Java and Swift. Mainly, developers has been focusing on the native building of applications because of the customization's that is enabled for the native devices such as camera access, etc.

4.1.1 Native implementation of UI

One thing that is associated with native mobile applications is that there are more fluid looking animations and easier integration with the mobile technology. Native applications inherit the targeted platforms looks and apply it to their appearance, this is called Native UI. This allows the native applications to behave and look more accordingly to the "native" system to give the user a more relatable experience to the mobile platform OS.

4.2 Cross-platform mobile application development

Cross-platforming refers to a product or software that can be used on another platform than what it was developed for. In app development the principle of cross-platforming is to build and maintain only one code base, which is the alluring part of using it since it saves time in development compared to native that is limited to only one platform per code base. Examples of cross-platform frameworks/tools are Flutter, React Native and Ionic.

4.3 Use and popularity

In a paper written by Mehdi Satei 2019, there is a section where the author discusses which programming languages and frameworks that are popular in the industry. Satei presents statistics from 2019 where Dart is on the rise of most wanted languages while Flutter is top 3 on most wanted frameworks.

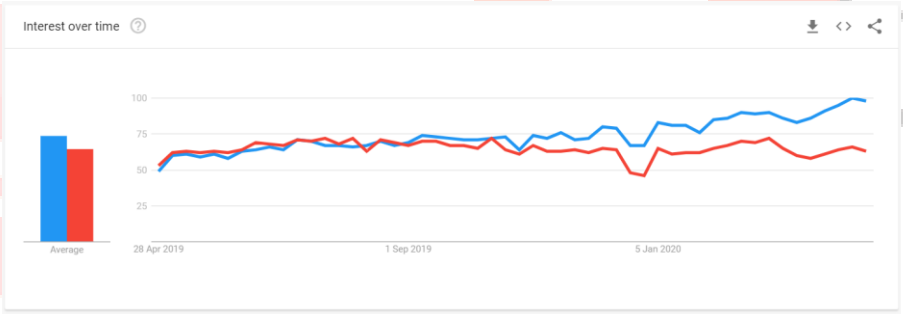


Figure 1: Trends of Flutter and React Native Usage

Looking at the Google trends chart for Flutter in comparison to React Native, there is a difference starting in trends for searching which means that more people are taking interest in Flutter development. Flutter is however, not used much when looking at the Stackoverflow survey of 2019 where Flutter is only used by 3.4% of all users and not even on the list for the professional developers use. 1.9% of the users liked Dart while for the professional developers use, Dart was not on the list. Both Flutter and Dart is however very highly ranked in the "Most loved" categories as 3rd and 12th.

In the Stackoverflow survey of 2020, Flutters use statistics has increased from 3.4% to 7.2% in the overall statistics, but stays off the list as before in the professional only statistics. Flutter stays in the 3rd position as the most loved framework/library/tool for 2020.

Dart is still off the list of the professional use of Dart, but has increased from 1.9% to 4.0% since 2019. For the 2020 Stackoverflow survey, it has become the 7th most loved programming language.

5. RESULTS OF BASIC APPLICATION

Looking at the results as a whole, Flutter wins the majority of most categories in the development area. There are however some differences that are interesting to take note of when comparing Flutter to native builds.

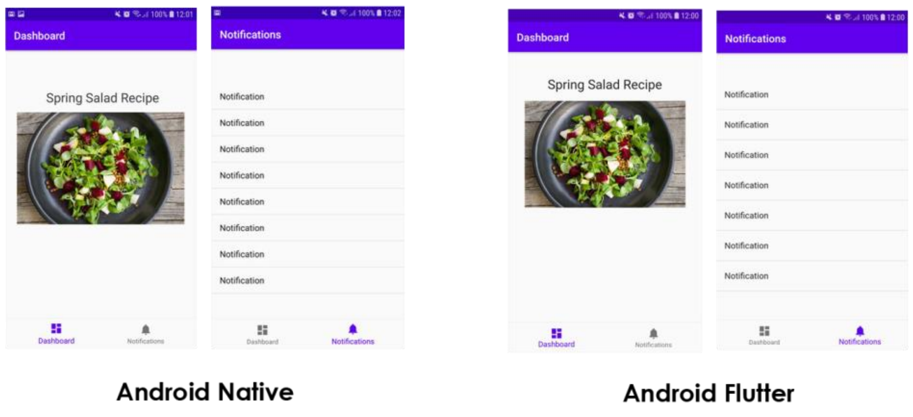


Figure 2: Images of Native Android and Flutter Applications

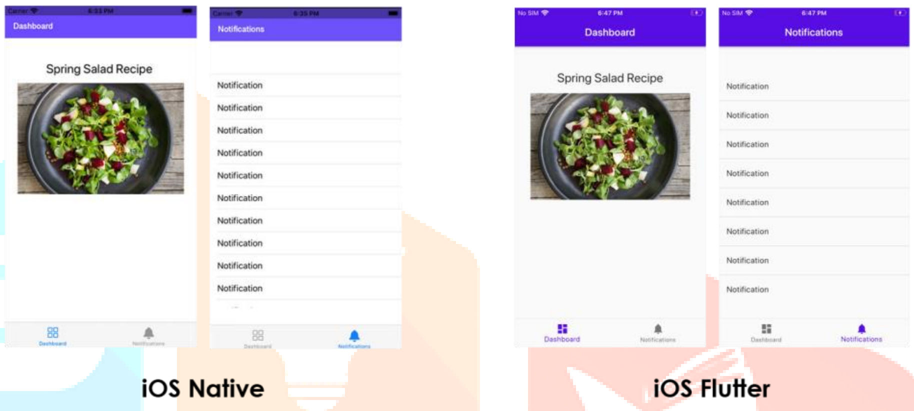


Figure 3: Images of Native iOS and Flutter Applications

Type	lines of code	Code files that were needed for the application
Android native	217	9
Android Flutter	125	3
iOS native	363	6
iOS Flutter	125	3

Figure 4: Application lines of code and file count

Type	Total	Navigation Base	First View	Second View
Android native	12h	3h	2h	7h
iOS native	8h	2h	0.5h	5.5h
Flutter	6h	4h	1h	2h

Figure 5: Development time of each code base

5.1 Code Size

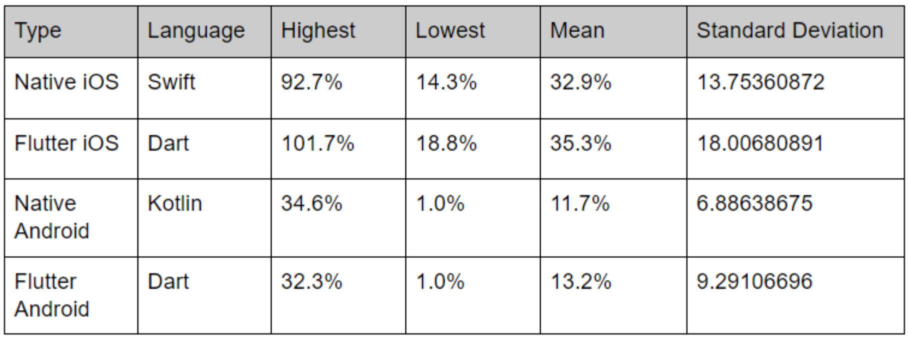
As seen in Figure 4, Flutter had the lowest amount of code lines and files that were needed in order to create the application. Native iOS had lower size of project files and app size but a significantly larger amount of code lines than the other builds. The native android had the most amount of files created and required lower amount of code lines than the iOS native.

5.2 Development Time

Android native had the longest developing time in total which was 12 hours, followed by the iOS native with 8 hours and lastly the Flutter which had the shortest of 6 hours.

In the development of both native applications, the use of drag and drop can be utilized for faster development. This is useful until a certain point when the dropped elements need to be connected with code which was harder to do than generate the layout through direct code. This makes it easier to develop because the fact that the layout is always visible without the need of building it. Corresponding development function in Flutter is the hot reload function which lets you build the application and be able to reload it based on new features added.

5.3 Flutter run time CPU performance comparison



Type	Language	Highest	Lowest	Mean	Standard Deviation
Native iOS	Swift	92.7%	14.3%	32.9%	13.75360872
Flutter iOS	Dart	101.7%	18.8%	35.3%	18.00680891
Native Android	Kotlin	34.6%	1.0%	11.7%	6.88638675
Flutter Android	Dart	32.3%	1.0%	13.2%	9.29106696

Figure 6: Table showing summarized results of CPU measurement

The highest CPU performance is more in Flutter iOS as compared to native iOS, while the Native Android has a better highest CPU performance than Flutter Android. But overall Flutter has a better mean CPU performance in both the OS so it beats the natives in both iOS and Android.

5.4 Application Looks

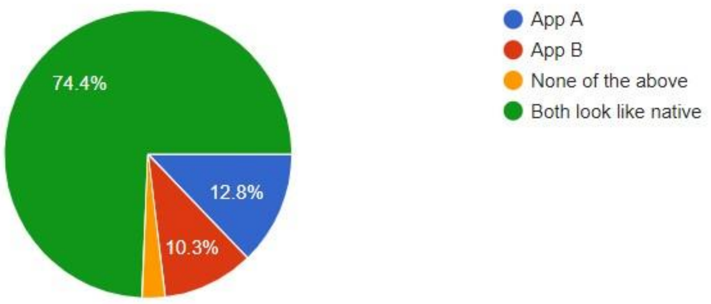


Figure 7: Diagram showing answers for looks of application Native (A) and Flutter (B)

This diagram is a survey conducted which clearly shows that most people believe that both the applications look the same.

6. CONCLUSION

Flutter is a useful toolkit that enables easy ways of creating new applications. It has gotten more and more popular recently. The basic results in this report indicates flutter has a slight edge as compared to native application development platforms but further more conclusive tests still needs to be carried out to come to a final conclusion.
Appearance wise, Flutter and native seem to differentiate little to a majority of users. It is able to mimic the native looks to a certain point.
To conclude the answers and ideas of Flutter, it is a tool with a promising feature if the community continues to grow in the direction that it is right now. The line to drawn when to choose Flutter over two separate native builds, can be chosen at the development of smaller to medium applications which are more flexible. Considering that Flutter’s strong side is being a cross-platform solution, Flutter still performs well on a single application base if compared to native applications. Flutter may not beat native for developing applications at this point but the results show good potential for the future although further studies needs to be done in these areas to conclude safer answers.

References

[1] Wenhau Wu. March 2018 Thesis React Native vs Flutter, Cross-Platform Mobile Application Framework.

[2] M. Satei, "Ott video-oriented mobile applications development using cross platform ui frameworks," KTH Royal Institute of Technology, School of electrical engineering and computer science, diva2:1343759: Diva, 2018, p. 90.

[3] C. Software. (2019). What is flutter? here is everything you should know, [Online]. Available: https : / / medium . com / @concisesoftware / what - is- flutter- here- is- everything- you- should- know- faed3836253f (visited on 03/05/2020).

[4] M. E. Joorabchi, A. Mesbah, and P. Krunchten, "Real challenges in mobile app development," University of British Columbia, University of British Columbia, 2013, p.10. [Online]. Available :http://citeseerx.ist.psu. edu/viewdoc/download?doi=10.1.1.724.2463&rep=rep1&type=pdf.

[5] C. Griffith. (2019) what is cross-platform app development, [Online]. Available: https://ionicframework.com/resources/articles/whatis-cross-platform-app-development.

[6] Google. (2019). Google flutter sdk releases, [Online]. Available: https://flutter.dev/docs/development/tools/sdk/releases

[7] W. Danielsson, "React native application developement," Diva, diva2:998793: Faculty of Computer science LIU, 2016

[8] Google. (2020). Material io components, [Online]. Available: https : / /material.io/components

[9] Stackoverflow, (2020). Stackoverflow survey 2020, [Online]. Available: https://insights.stackoverflow.com/survey/2020

[10] G. Flutter. (2020). Google flutter animations, [Online]. Available: https://flutter.dev/docs/development/ui/animations