Comparison of Flutter and React Native Platforms

Article · December 2021

DOI: 10.34231/iuyd.888243

3 authors:

Ekrem Gülcüoğlu
Bartin University
2 PUBLICATIONS **2** CITATIONS
SEE PROFILE

Ahmet Berk Ustun
Bartin University
34 PUBLICATIONS **177** CITATIONS
SEE PROFILE

Neşet Seyhan
Bartin University
3 PUBLICATIONS **1** CITATION
SEE PROFILE

Hakemli - Süreli - Bilimsel Dergi
İNTERNET UYGULAMALARI ve YÖNETİMİ
Journal of
INTERNET APPLICATIONS and MANAGEMENT
2021 / 12(2)

Comparison of Flutter and React Native Platforms

Flutter ve React Native platformlarının karşılaştırılması

Ekrem GÜLCÜOĞLU [1], mehterli@gmail.com
Ahmet Berk ÜSTÜN [2], ustun.ab@gmail.com
Neşet SEYHAN [3], nesetseyhan@gmail.com

Received: 28.02.2021; **Accepted**: 29.12.2021

Mobile application development processes have become very important with the increasing use of mobile devices. Android and IOS operating systems, which are the most preferred mobile operating systems, have different development tools and software languages. Developing and testing and debugging the same application separately for each platform with development tools can lead to time loss and high maintenance costs. Cross-platform software that provides testing and development for different operating systems at the same time offer an alternative structure to these processes. Cross-platform software accelerates software processes and provides advantages by reducing application development costs. The comparison of Flutter and React Native, which are cross-platform development software will be presented in this study.

Keywords: *Mobile Applications, Flutter, React Native, Cross Platform*

Artan mobil cihaz kullanımı ile birlikte mobil uygulamaları geliştirme süreçleri de oldukça önemli hale gelmiştir. Mobil işletim sistemlerinden en çok tercih edilen Android ve IOS işletim sistemleri farklı geliştirme araçları ve yazılım dillerine sahiptir. Geliştirme araçları ile her platform için ayrı olarak, aynı uygulamayı geliştirmek ve test ederek yayınlamak zaman kaybına ve yüksek bakım maliyetlerine yol açabilmektedir. Farklı işletim sistemleri için aynı anda test ve geliştirme olanağı sağlayan çapraz yazılım platformları bu süreçlere alternatif yapı sunmaktadır. Çapraz yazılım platformları yazılım süreçlerini hızlandırırken, uygulama geliştirme maliyetlerini düşürerek avantajlar sağlamaktadır. Bu çalışmada çapraz yazılım geliştirme platformlarından olan Flutter ve React Native'in karşılaştırılması sunulacaktır.

Anahtar Kelimeler: *Mobil Uygulamalar, Flutter, React Native, Çapraz Platform*

[1] Tosya Mithat Boyner Vocational and Technical High School (Corresponding Author)
[2] Bartın University, Computer Technology and Information Systems
[3] Bartın University, Department of Information Technologies

1. INTRODUCTION

There are many options to develop mobile applications that have become an important part of our lives nowadays (Keskin & Kılınç, 2015). Although there are different mobile operating systems, basically the most preferred mobile operating systems are Android and IOS operating systems (Goadrich & Rogers, 2011). Development tools of these platforms are Android Studio and Xcode (Vilček & Jakopec, 2017). Xcode and Android Studio development tools differ from each other in that they support different software languages and have different interfaces. Java or Kotlin languages are used for Android development, while Objective-C or Swift software languages are used for IOS. It can take a lot of time to use these software languages and adapt to these development environments. Developing and testing the same application by separately using these development tools for each platform can result in a lot of time loss and high maintenance costs (Kaur & Mishra, 2019).

Cross-platform methods that emerged to support all platforms with a single code for an application created using different tools have become quite popular for this reason (Yatsenko vd., 2019). Cross-platform support is used for applications that can run on at least two operating systems. Xamarin, Ionic, Adobe Phonegap, React Native, Flutter development platforms are examples of cross-platform methods (Gerasimov, Bilovol, & Ivanova, 2015; Packer vd., 2019; Yatsenko vd., 2019).

Xamarin is a cross-platform structure for developing mobile applications using the C# language with the Visual Studio development kit. The company was founded with the same name in 2011 and acquired by Microsoft in 2016. It offers developers a wide variety of tools that can be used for cross-platform mobile application development, it is widely used today, especially for Android applications. (Javatpoint, 2020). The World Bank, Storyo were Developed by APX Xamarin. It is a cross-platform development method based on Ionic, Angular Js and Apache Cordova. It was first used in 2013 (Devnot, 2020). It is possible to develop hybrid applications with open source ionic using HTML and Javascript. It allows development for both IOS and Android operating systems. Sworkit, Justwatch, Mclaren Automotive, Diesel, Chefsteps are some of the applications developed with Ionic coding. It works with Adobe Phonegap, HTML, CSS and Javascript technologies. It is developed on Apache Cordova architecture. It is a cross-platform application development platform first founded by Nitobi company in 2011 and later acquired by Adobe (PhoneGap, 2012). It is one of the open source mobile development platforms. Some of the applications developed with Phonegap include Wikipedia, Tripcase, Healthtap. Flutter allows for creating cross-platform applications. It is also a software development tool that can output close to Native for Android and IOS (Zammetti, 2019). It is a software development platform that is being developed by Google acquired without any fee and whose popularity is increasing day by day (Trends, 2020b). React Native is a Javascript-based structure that enables cross-platform application development for mobile devices and can execute close to Native like Flutter. Although it is open source and free, an application can be developed for IOS and Android with a single coding with state management and with plugins (Eisenman, 2015).

Popular applications shared in Flutter's development library are The New York Times, Alibaba, Google Ads, eBay, Groupon, Tencent, Square, Google Assistant, Baidu, Philips Hue, Groupon (Dev, 2020). The main popular applications shared in React Native's development

library are Facebook, Facebook Manager Ads, Facebook Analytics, Instagram, Skype, Oculus, Uber Eats, Walmart, Pinterest, Tesla, Wix.com, Tencent QQ, Adidas, GLITCH (Development, 2020). In this study, React Native and Flutter two methods which are supported by Google and Facebook, among the above-mentioned development methods which have recently become very popular, were compared. Basic information about the platforms will be shared in the first part; the languages and tools used will be compared in the second part. The advantageous features and missing aspects of these two platforms will be examined in the third part.

2. THEORETICAL FOUNDATION AND HYPOTHESES DEVELOPMENT

2.1. React Native

React Native was first introduced in 2015 and it continues to be developed (Native, 2020). In 2015, this rapidly developing open source platform was added to the favorite list by over 30,000 programmers on GitHub. (Team, 2020). This platform, based on Javascript, was built on the React framework created for the development of web pages and applications can be developed for both IOS and Android operating systems. (Occhino, 2020). The design scheme is specified in Figure 1.

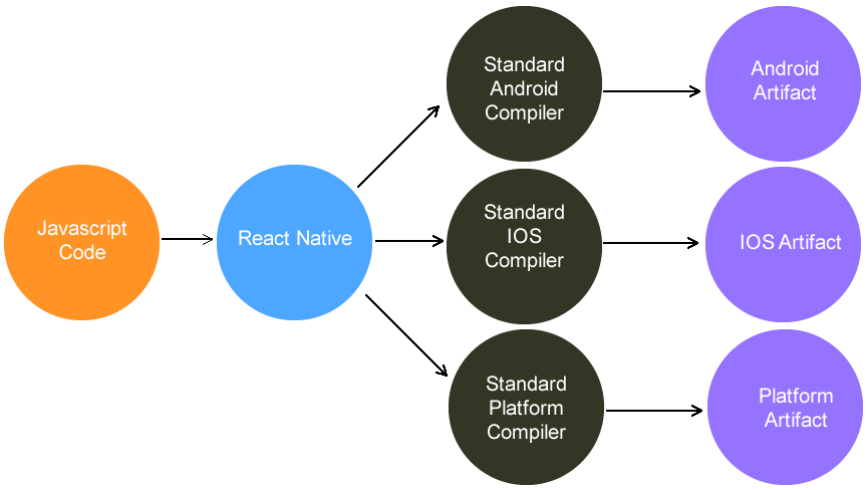

Figure 1. Structural Framework of React Native (Frachet, 2020)

A programmer can start developing a new application with any javascript supporting IDE (Integrated development environment). Many IDEs have javascript support. Applications are tested with support from the simulators of third source software in the development environment such as expo, IOS and Android.

2.2. Flutter

Flutter was introduced at "Google Developer Days" in 2018. It is an open source and free mobile development SDK developed and supported by Google, used to develop apps for Android and IOS, as well as the primary method of building apps for the Google Fuchsia operating system (Developers, 2018; Singh & Bhardwaj, 2019). Flutter is built in C / C ++ in Dart languages and uses the Skia Graphics Engine. The design scheme is specified in Figure 2. It is using Dart programming language. Applications are developed with objects called widgets. Applications with "Material Design" and "Cupertino" objects can be compiled for

both Android and IOS operating systems at once without changing the platform independent emulator. With its popularity on GitHub where more than 100 million open source projects are shared, it has an extremely high tendency among mobile software developers (GitHub, 2020). Data in an application can be checked by means of state management (Kuzmin, Ignatiev, & Grafov, 2020).

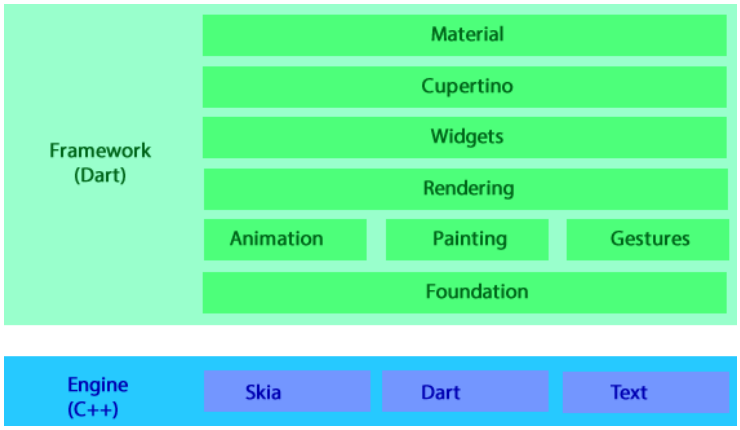

Figure 2. Flutter's structural framework (Architecture, 2020)

3. DATA ANALYSIS AND RESULTS METHODOLOGY

3.1. Interest of Developers

In Stackoverflow (Vasilescu, Filkov, & Serebrenik, 2013), a forum site where software developers exchange ideas with each other, discuss and ask questions, the result for the programming language they like according to the interest of the programmers is shared in Figure 3.

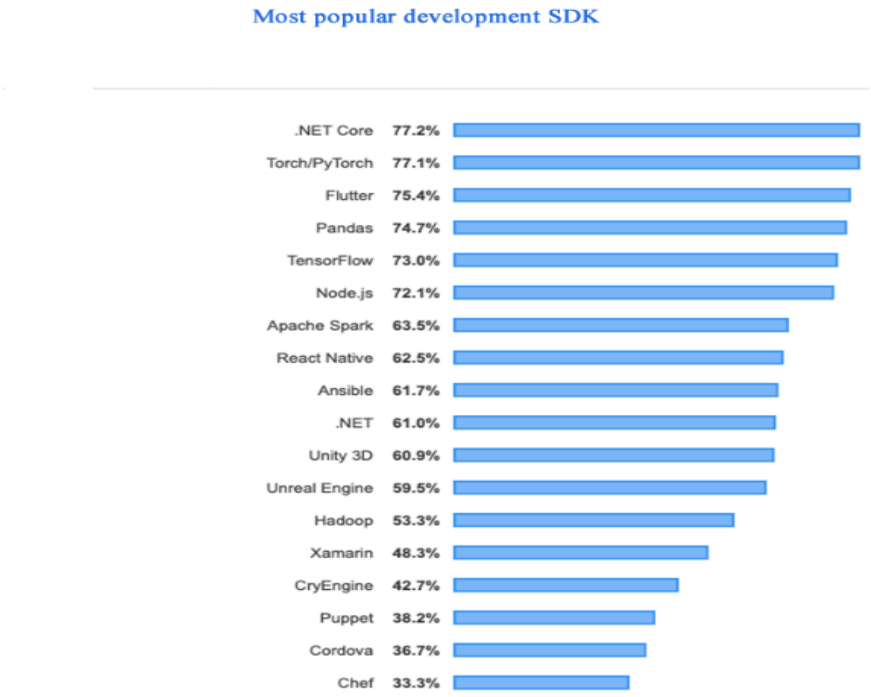

Figure 3. Ranking among the most popular programming languages (Stackoverflow, 2019)

According to the ranking result in Figure 3, Flutter, React Native and Xamarin, which are among the cross-platform structures, have entered the list. While Flutter ranked first with 75.4% among cross-platform structures, React Native ranked second among these platforms with 62.5% and made a big difference to Xamarin, which ranked third with 48.3%. In addition, according to this list, these two platforms among other structures were among the top 10.

The graph of the search for React Native and Flutter in all categories on Google.com in the last 12 months is shared in Figure 4.

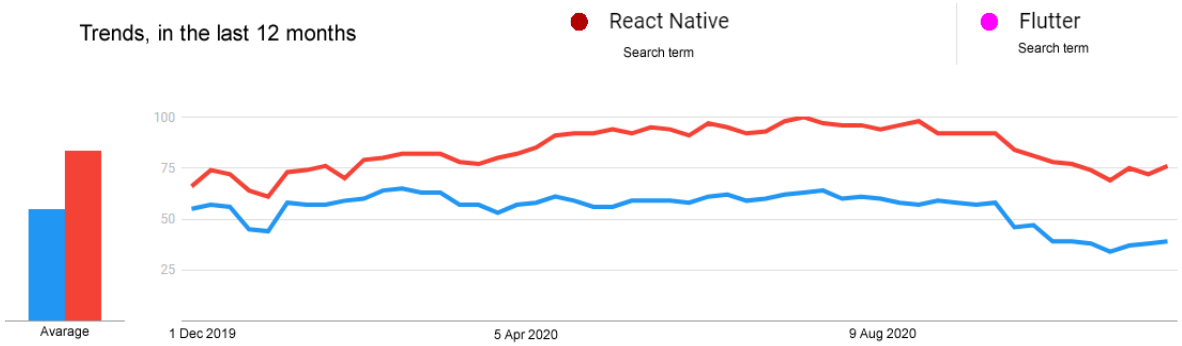

Figure 4. React Native vs Flutter search comparison on Google (Trends, 2020b)

The interest shown in Flutter and React Native in the last year is shared in Figure 4. According to this, the numbers show search interest globally in the last year relative to the highest point on the chart. A value of 100 is that the term has the highest popularity. A value of 50 means that the term is half as popular as. A value of 0 means that there is not enough data for this term. While the search frequencies of Flutter and React Native have been close to each other since September 2019, the search frequency of Flutter has increased more in the last 9 months. Ups and downs are observed in the React Native graph. In parallel with the React Native graph, Flutter graph that has an increasing search frequency draws attention.

Comparing the tags opened on the Stackoverflow website; The result of Ionic, React Native, Flutter and Xamarin comparison is as in Figure 5. There is an average of 1500 questions difference between Flutter and its closest follower, React Native. Although Flutter was released 2 years after React Native, it quickly closed the gap with the increase in the number of users it gained and entered the race.

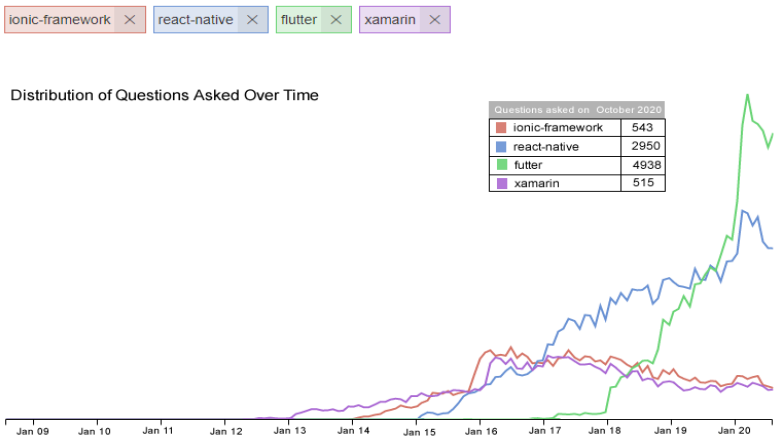

Figure 5. Stackoverflow Tag Benchmark (Trends, 2020a)

3.2. Structure and Concepts in Flutter

Each structure is considered as a class and applications are developed with object-oriented programming in Flutter. Classes are associated with objects defined as "Widgets" by taking values (property) defined in Flutter's own library.

```
Scaffold(
body: Center(
child: Text(
"Merhaba dünya",
style: TextStyle(
color: Colors.blue,),),);
```

Figure 6. Sample code in Flutter

The body object (Widget) is created from the Center class as indicated in Figure 6. It is associated with the Text (Widget) object while passing Center class to the child property. However, the "Hello world" expression in the page title has been transferred to the Text object as a value. By assigning the TextStyle object (Widget) to the style property of the Text object; widget-class-property structures in nested form, applications can be developed within an object-oriented framework. (Payne, 2019). There are hundreds of objects such as Body, Text, Image, Row, Column, Icon, Scaffold, Container. (Cheng, 2019).

3.3. Using Variables

It contains two extra features in addition to the variable definition used in the classical programming approach in terms of variable definition.

3.4. Var Using

This data type has no specific value. The desired variable can be assigned bool, integer, char and so on. Example usage is indicated in Figure 7.

```
Var isim= "Mehmet";
Var mesaj= "Hoş geldin $isim";
```

Figure 7. Sample code for var

3.5. Dynamic Usage

This data type has a variable structure. The variable type created with Dynamic is first assigned an integer type value. During use, the value can be transferred to the string data type later (data types such as string, the integer is exemplified because they are commonly used data types. This is valid for all data types supported by Dart.) Example usage is indicated in Figure 8.

```
Dynamic bilgi="string";
bilgi=5;
```

Figure 8. Sample code for dynamic

3.6. Control and Loop Structures

While commonly used control and loop structures such as if, if else, while, do can also be used in dart language, they support the "ternary if" structure since it was developed from the C programming language. In Figure 9, classical if usage and in Figure 10 an example of using "ternary if" are shared.

```
String değer= "";
if 1>0 deger = "büyük" else değer="küçük"; print(değer);
```

Figure 9. Classic if usage

```
String değer= 1>0 ? "büyük" : "küçük"; print(değer);
```

Figure 10. Usage of ternary if

3.7. Lists

There is no array-like structure definition in Dart language, instead list structure is used. The list structure is basically used in 2 different ways.

1. Fixed length
2. Dynamic structure

The list structure can contain variable data types according to the assigned value and must be sequential. In addition, it includes ready-made methods for performing operations such as adding and deleting data. An example of using List is given in Figure 11.

```
// Sabit boyutlu List kullanımı
List<int>  bilgi=new List(3);
List<int> bilgi2=[10,100,1000];
// Dinamik yapıda list kullanımı
List<String> harfler=new List();
```

Figure 11. Usage of list

3.8. Set Structure

It is used to store more than one data like list and has the same properties as list. The difference from list is that it is unordered and does not contain 2 identical elements (Unique). The set and list definition code sample are shared in the Figure 12. The screenshot after sample run is presented in the Figure 13.

```
Set<String> bilgi=Set();
List<String> data=new List();
Main()
{ bilgi.add('Bilgi');
bilgi.add('Bilgi');
bilgi.add('Bilgi');
bilgi.add('Bilgi2');
data.add('Data');
data.add('Data');
data.add('Data1');
data.add('Data2');

print("Set kullanımına dair oluşan yapı:"); print(bilgi);
print("List kullanımına dair oluşan yapı:"); print(data);
```

Figure 12. Usage of set and list

```
Set kullanımına dair oluşan yapı:
{Bilgi, Bilgi2}
List kullanımına dair oluşan yapı:
[Data, Data, Data1, Data2]
```

Figure 13. Output about using set and list

3.9. Usage of Lambda Functions

Every function in the Dart language is also an object. Frequently used functions that have no names are also called lambda functions. Example usage is shown in Figure 14. The screenshot of the sample code is run is shared in Figure 15.

```
main()
 {printDeneme();
 print(deneme1());
 print(deneme2());
 }
printDeneme()==>print("selam");
int deneme()=>100;
bool deneme2{return false;}
```

Figure 14. Usage of Lambda

```
selam
100
false
```

Figure 15. Output of Lambda function

3.10. Substructure

Flutter is an SDK that offers reactive programming (Cosmina, 2020). Its objects through communication with local hardware do not need another library. Dart code is compiled directly into local machine code. This situation is indicated in Figure 16.

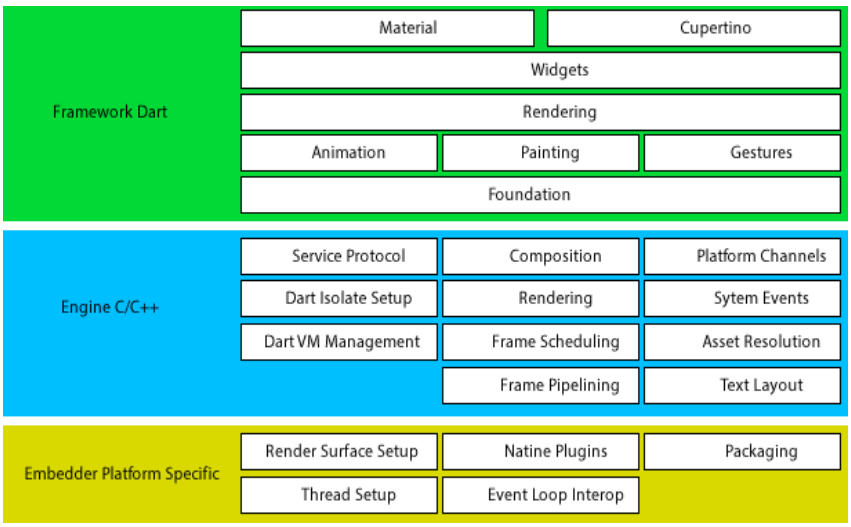

Figure 16. Flutter Running Architecture (Architecture, 2020)

3.11. Structure and Concepts in React Native

While developing applications with React Native, structures called components are used. Components such as Text, View, Button, Image, TextInput can be given as examples of these structures. These components can be customized just as they were designed while developing the web interface. A sample React Native code block using View, text, image, scrollview, textinput objects is shared in Figure 17.

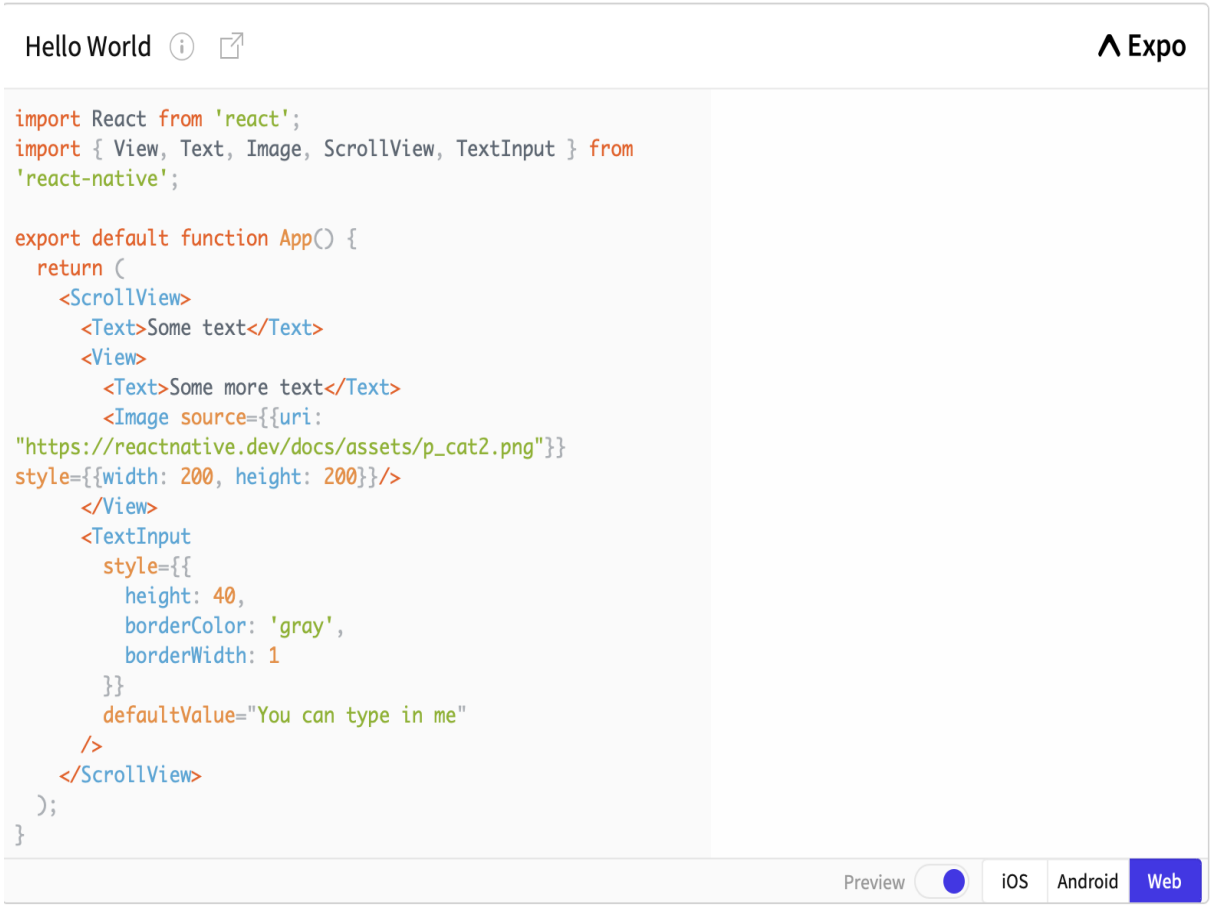

Figure 17. Sample code for React Native

The creation and arrangement of components are structurally similar to HTML. For example, View component has a similar structure with "div" tags in Html. These components can be customized just like Html. Libraries of these customized components can also be used in different projects.

Component equivalents of the basic React Native components by platforms are given in Table 1 below.

Table 1. Comparing React Native components with other platforms

React Native Components	Android	IOS	Web	Description
<View>	<ViewGroup>	<UIView>		Style is the non-scrolling component that includes other components that support editing accessibility controls.
<Text>	<TextView>	<UITextView>	<textarea>	It is a styled component that can be added to show text.
<Image>	<ImageView>	<UIImageView>	<img>	For displaying different types of images.
<TextInput>	<EditText>	<UITextField>	<input type="text">	It is a component that provides text input to the user.
<ScrollView>	<ScrollView>	<UIScrollView>		Style is the scroll-enabled component that includes other components that support editing accessibility controls.

3.12. Using Variables

React Native is not structurally a software language. It is basically a framework created by Javascript. Therefore, the use of variables is similar to Javascript. Within React Native functions, variables can be assigned similarly to the Flutter by using both "var" and "let" structures. The difference between var and let is that variables defined with let can only be accessed within the block they are in. There is an example of let usage in Figure 18.

```
var mesaj = "Merhaba";
let text = "Merhaba";
```

Figure 18. Usage of Var and Let

Apart from var and let there is also a const. It is not possible to assign a variable to a variable created with Const. Figure 19 shows an example of const usage.

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    paddingTop: Constants.statusBarHeight,
    backgroundColor: '#ecf0f1',
    padding: 8,
  },
  paragraph: {
    margin: 24,
    fontSize: 18,
    fontWeight: 'bold',
    textAlign: 'center',
  },
});
```

Figure 19. Example usage of the Const

State structure is used to store data in React Native components, except let, var, const, which are used for variable definitions.

3.13. Arrow Functions

The standardized version of the javascript language called ECMAScript is used in React Native projects. With ECMAScript 6, arrow functions called arrrow functions, which are alternative to functions, have been added. Different from the standard functions, the expression in Figure 20 is used instead of the function expression used in each definition in arrow functions. Arrow functions can be useful if the function content will not be used after processing the line found.

```
() => {}
```

Figure 20. Arrow Function

3.14. Setup

First of all, by installing Android Studio or Visual Studio Code for the installation of Flutter; The preferred editor for the coding interface is loaded. In the next step, the installation is completed by loading the Flutter components into the editor. There is a module with the given name Flutter Doctor, deficiencies or errors in the installation program can be viewed at any time from the terminal.

In order to develop applications in React Native, it is possible to start developing and testing online on the "expo.snack.io" web page without any installation. Apart from this, by installing the necessary React Native components, development can be made on Windows, Linux and Mac operating systems using Android and IOS simulators for testing.

For example, project creation and initialization are as follows.

React Native

$ npx react-native init yeniUygulama

$ cd yeniUygulama

3.15. Support and Communities

Flutter has been following an increasing trend on Google and Github in a short 2-year period since 2018 (GitHub, 2020; Trends, 2020b). It is open source and distributed without any fee. The developer of Android is powered by Google.

1. There are tens of thousands of questions and answers in its own help library on GitHub, Stackoverflow. Questions are exchanged and various methods are discussed between developers and beginners (GitHub, 2020).

2. There are more than 980,000 students and 140 courses for Flutter on Udemy (Courses, 2020).

3. Flutter works gained popularity among developers in less than 2 years, getting over 93,200 stars on Github (Issues, 2020).

4. Flutter provides support in its own library and website.

3.16. Common Features of Flutter and React Native

1. They can be developed easily in Windows, Linux or MAC environment. It is sufficient to have any of these 3 operating systems installed on the desktop or laptop computer where it will be installed.

2. They have a rich widget library used in Android-IOS platforms by supporting the modern react-style structure with mobile-first support (Flutter, 2018).

3. It has a real-time compilation feature called "Hot Reload" that is not available in many development environments. With "Hot Reload", the developer is designing or coding the application; the changes made in real-time can be viewed on the application (Payne, 2019).

As a result of the researches conducted within the scope of the study, the results of the comparison of Flutter and React Native, which are cross-software development platforms, based on 6 different titles are presented in Table 2.

Table 2. Flutter vs React Native Comparison

	Flutter	**React Native**
Software Languages Used	Dart	Javascript
Development Environment	Windows, Mac OS, Linux	Windows, Mac OS, Linux, Online
Hot Reload Support	Supported	Supported
Supported Operating Systems	Android, Ios and Windows	Android, Ios and Windows
3rd Party Library Software Support	A few	A lot
Simple 'Helle World' applications size	4.7 MB	7 MB

4. DISCUSSION AND CONCLUSION

As a result of this research, it has been determined that both platforms have useful aspects, any platform does not have a predominant advantage over the other. The number of users of these two open source applications is increasing day by day (Trends, 2020b). It is seen that React Native's development environment is more advantageous compared to Flutter in the areas of preparation of the development environment, online application development, editor diversity, the ability to use the platform without learning a new language for those who know the Javascript language, diversity for ready-to-use components and accessible resources. It is clear that Flutter's three-stage test system is more useful in terms of the application size taking up less space, being faster at the first opening of the application, and increasing the usage trends of the developers. Although they have structural differences, it is possible to develop mobile applications for Android and IOS with these two platforms. While developing applications on both sides, making use of the "Hot Reload" feature provides both time saving and functionality for the developer. Both applications support all currently used operating systems. The dimensions of the applications created according to native applications on both platforms are more. However, considering the ability to develop both IOS and Android

applications at the same time on both platforms, the size of the applications created can be ignored.

Considering the popularity of Javascript in today's world, regarding the difficulties or problems encountered while learning a language or developing an application. Responses to the questions asked on the software forums are faster in React Native and it is ahead of Flutter in this regard (Stackoverflow, 2020). The Dart language used in Flutter (for someone who knows at least one of the object-oriented programming languages such as C #, C ++ or Java) is very easy to learn. React Native uses the JavaScript language known and used by many users. React Native is highly dependent on third-party software. When developers want to use native modules (Bluetooth, Wi-Fi, SMS, etc.), they may have to use third-party libraries. On the Flutter side, dependency on third-party software is less on to the widgets (device API access, navigation, etc.) (Cheng, 2019). In terms of corporate support, React Native has been supported by Facebook since 2015 (Eisenman, 2015). Flutter has been supported by Google since 2018 and continues to develop like React Native (Boukhary & Colmenares, 2019). They do not have any major advantages or weaknesses against each other in terms of institutional support. As a result, it is not possible to talk about a significant superiority between these two-cross platform development software. These two development environments, which are renewed day by day and increase the number of users, can be preferred by programmers. With these two development environments, it is possible to produce applications for different platforms by saving time.

REFERENCES

Architecture, F. (2020). *Flutter Architecture.* Retrieved from: https://www.javatpoint.com/flutter-architecture Access Date: 28/11/2020.

Boukhary, S., & Colmenares, E. (2019, December). *A Clean Approach to Flutter Development through the Flutter Clean Architecture Package.* IEEE, International Conference on Computational Science and Computational Intelligence (CSCI). Doi: https://doi.org/10.1109/CSCI49370.2019.00211.

Cheng, F. (2019). *Flutter Recipes.* CA: Apress.

Cosmina, I. (2020). *Building Reactive Applications Using Spring.* CA: Apress.

Courses, U. (2020). *Google Flutter Courses.* Retrieved from: https://www.udemy.com/topic/google-flutter/ Access date: 28/11/2020.

Dev, F. (2020). *Flutter Developer Library*: Retrieved from: https://flutter.dev/showcase Access date: 28/11/2020.

Developers, G. (2018). *Google Developer Day at GDC 2018 Livestream.* Retrieved from: https://www.youtube.com/watch?v=5wtlj_q3DjE&list=PLOU2XLYxmsIIxxDKHWd_aId_oV9hqPi7q Access date: 28/11/2020.

Development, R. N. (2020). *React Native Developer Library*: Retrieved from: https://reactnative.dev/showcase Access date: 28/11/2020.

Devnot. (2020). *Ionic'i Tanıyalım.* Retrieved from: http://devnot.com/2016/hibrit-uygulama-catisi-ionic-i-taniyalim Access date: 28/11/2020.

Eisenman, B. (2015). *Learning React Native: Building Native Mobile Apps with Javascript.* CA: O'Reilly Media Inc.

Flutter (2018). *Technical Overview.* Retrieved from: https://flutter.dev/docs/resources/technical-overview Access date: 28/11/2020.

Frachet, M. (2020). *Understanding The React Native Bridge Concept*: Retrieved from: https://hackernoon.com/understanding-react-native-bridge-concept-e9526066ddb8 Access date: 28/11/2020.

Gerasimov, V., Bilovol, S., & Ivanova, K. (2015). Comparative Analysis Between Xamarin and Phonegap For .Net. *System technologies, 96*.

GitHub. (2020). *A Small Place to Discover Languages in Github.* https://madnight.github.io/githut/#/issues/2020/2 Access Date: 28/11/2020.

Goadrich, M. H., & Rogers, M. P. (2011). *Smart Smartphone Development: Ios Versus Android.* Paper presented at the Proceedings of the 42nd ACM technical symposium on computer science education, New York: USA.

Issues, G. F. (2020). *Want To Contribute to Flutter / Flutter?* https://github.com/flutter/flutter/issues Erişim tarihi: 28/11/2020.

Javatpoint. (2020). *What is Xamarin*: Retrieved from: https://www.javatpoint.com/what-is-xamarin Access date: 28/11/2020.

Kaur, L., & Mishra, A. (2019). Cognitive Complexity as A Quantifier of Version to Version Java-Based Source Code Change: An Empirical Probe. *Information and Software Technology, 106*, 31-48.

Keskin, N. Ö., & Kılınç, H. (2015). Mobil Öğrenme Uygulamalarına Yönelik Geliştirme Platformlarının Karşılaştırılması ve Örnek Uygulamalar. *Açıköğretim Uygulamaları ve Araştırmaları Dergisi, 1*(3), 68-90.

Kuzmin, N., Ignatiev, K., & Grafov, D. (2020). *Experience of Developing a Mobile Application Using Flutter.* (pp. 571 - 575). Springer International Publishing. Doi: https://doi.org/10.1007/978-981-15-1465-4_56.

Native, R. (2020). *React Native* Retrieved from: https://reactnative.dev Access date: 28/11/2020.

Occhino, T. (2020). *React Native: Bringing Modern Web Techniques to Mobile*: Retrieved from: https://engineering.fb.com/android/react-native-bringing-modern-web-techniques-to-mobile/ Access date: 28/11/2020.

Packer, C., Wade, T., Diaz, M., Habjan, A., McIntyre, R., & Ancona, B. (2019, April). *Implementing a mobile application for fraternity and sorority life.* (784). Symposium on undergraduate research and creative expression (source), Valparaiso.

Payne, R. (2019). *Beginning App Development with Flutter: Create Cross-Platform Mobile Apps.* CA: Apress, Retrieved from https://doi.org/10.1007/978-1-4842-5181-2_2

PhoneGap. (2012). *PhoneGap Beliefs, Goals, And Philosophy*. https://blog.phonegap.com/phonegap-beliefs-goals-and-philosophy-dc9d1f7d7aca Erişim tarihi: 28/11/2020.

Singh, T., & Bhardwaj, R. (2019). Fuchsia OS-A Threat to Android. *IITM Journal of Management and IT, 10*(1), 65-67.

Stackoverflow. (2019). *Developer Survey Results.* Retrieved from: https://insights.stackoverflow.com/survey/2019 Access date: 28/11/2020.

Stackoverflow. (2020). *Stack Overflow Trends.* Retrieved from: https://insights.stackoverflow.com/survey/2019#technology Access date: 28/11/2020.

Team, W. A. (2020). *React Native on the Universal Windows Platform*: Retrieved from: https://blogs.windows.com/windowsdeveloper/2016/04/13/react-native-on-the-universal-windows-platform/ Access date: 28/11/2020.

Trends. (2020a). *Tag Trends.* http://sotagtrends.com/?tags=ionic-framework+react-native+flutter+xamarin&relative=false Access Date: 28/11/2020.

Trends. (2020b). *React native - flutter comparison.* (d). https://trends.google.com/trends/explore?q=React%20Native,Flutter Access Date: 28/11/2020.

Vasilescu, B., Filkov, V., & Serebrenik, A. (2013). Stackoverflow and Github: Associations Between Software Development and Crowdsourced Knowledge. *2013 International Conference on Social Computing*, 188-195.

Vilček, T., & Jakopec, T. (2017). *Comparative Analysis of Tools for Development of Native and Hybrid Mobile Applications.* Paper presented at the 2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Rijeka: Croatia.

Yatsenko, R., Obodiak, V., & Yatsenko, V. (2019). Comparative Analysis of Cross-Platform Frameworks for Mobile Applications Development. *The Art of Scientific Mind,* (4), 132-136.

Zammetti, F. (2019). *Flutter: A Gentle Introduction.* APress, CA: Apress.