

## Comparative Analysis of Native and Hybrid Applications and Their Future

Abhishek Shukla

USA

### ABSTRACT

Due to the increment in technology, various mobile applications have become an integral part of our daily lives because users rely on them for different purposes from entertainment and communication to e-commerce. Therefore, there are two main approaches used for developing mobile applications have emerged the first one is hybrid and the second one is native applications. This article will provide a comprehensive comparative analysis based on these two approaches by considering some factors like development time, cost, user experience, performance, and the evolving landscape of mobile technology. Furthermore, it will also explore the prospects of these native and hybrid applications in detail by examining how emerging trends and technologies are impacting popularity and relevance.

### \*Corresponding author

Abhishek Shukla, USA.

Received: December 02, 2022; Accepted: December 09, 2022; Published: December 19, 2022

**Keywords:** Mobile Application, Android, iOS, Hybrid Applications, Native Application, Comparative Analysis, Performance, Cost, Development Time, Future Trends, User Experience

### Introduction

With the introduction of mobile applications, the interaction with technology has been revolutionized and its importance level in our daily life is increased. now it is extremely difficult to exclude it from daily life. Most individuals use it for entertainment and some of them perform daily routines through it like managing finances or staying connected with its customers. However, all tasks are performed through mobile applications so they are considered an integral part of modern society. To meet the growing demand for mobile applications, different developers have adopted different approaches. Therefore, with hybrid and native application development the development approach has been changed [1].

On the other hand, the hybrid applications are built through web technologies including JavaScript, CSS, and HTML, and then wrapped in a native container for deployment across various platforms [2].

This journal aims to conduct a proper analysis of hybrid and native applications by shedding light on their strengths, prospects, and weaknesses. By understanding the relative merits of these approaches, businesses and developers can make informed decisions when choosing the suitable method for mobile application projects.

### Literature Review

Various authors have discussed various facts about native and hybrid applications and their characteristics, including

performance, cost, development time, user experience, and future trends [2].

### Performance

All the native applications are known for their superior performance. These applications can leverage the full potential of the underlying operation, and hardware that will result in fast load times and smooth user experience. Furthermore, in hybrid applications, there are some problems related to performance. These applications contain performance bottlenecks because of the intermediary layer present between the device's native framework and code. Secondly, some latest advancements in hybrid frameworks like Flutter and react native have narrowed this performance gap and made the hybrid applications extremely competitive in terms of speed and responsiveness level [3].

### Development Time

For the development of Native applications, there is a need for separate codebases used for each platform and these codebases can easily increase the development time and cost. On the other hand, hybrid applications offer cross-platform compatibility that allows the developers to write code once and deploy it on different platforms. It will minimize the development time and simplify maintenance. For complex features and platform-specific optimization, there is still a need for additional efforts in the development of hybrid applications [1].

### Cost

It shows that the upfront cost for the development of native applications is extremely high. The main reason behind this is that there is a need for separate development teams and codebases for each platform. Moreover, hybrid applications are extremely cost-effective because they can leverage a single codebase for

different platforms which will minimize the development and maintenance expenses [1].

### User Experience

The native applications provide an intuitive and seamless experience to individuals because they are designed for platform-specific guidelines and can access device-specific features seamlessly. On the other hand, the hybrid applications may struggle to replicate this level of user experience. However, they have made some important progress in the last few years by improving libraries and frameworks [2].

### Future Trends

In the future, mobile applications will be developed in such a way that they can be shaped by various emerging technologies like virtual reality, augmented reality, and the Internet of Things. However, some native applications may continue to excel in various domains because of close integration with platform-specific APIs and hardware. Furthermore, hybrid frameworks are evolving to support these technologies and make them extremely competitive in the long term [2].

Layer	Android	iOS
1	Applications	Cocoa Touch
2	Application framework	Media layer
3	Libraries	Core services
4	Linux Kernel	OS layer

**Figure 1:** Layer Hierarchy for Android and iOS

In the above figure, there is some information regarding the layer hierarchy for iOS and Android in detail. It shows that there are four main layer hierarchies present for both operating systems. It can be observed that the first layer of Android is applications, and for iOS is Cocoa Touch. The second layer for Android is the application framework, and for iOS is the media layer. The third layer of Android is Libraries and for iOS is core services. The last layer for Android is Linux Kernel and for iOS is the OS layer [1].

	Web	Hybrid	Interpreted	Generated
Marketplace deployment	No	Yes, but not guaranteed*	Yes**	Yes**
Widespread technologies	Yes	Yes	Yes	No
Hardware and data access	Limited	Limited	Limited	Full access
User interface and look & feel	Simulated	Simulated	Native	Native
User-perceived performance	Low	Medium	Medium	High

**Figure 2:** Comparative Analysis for Cross-Platform Development Approaches

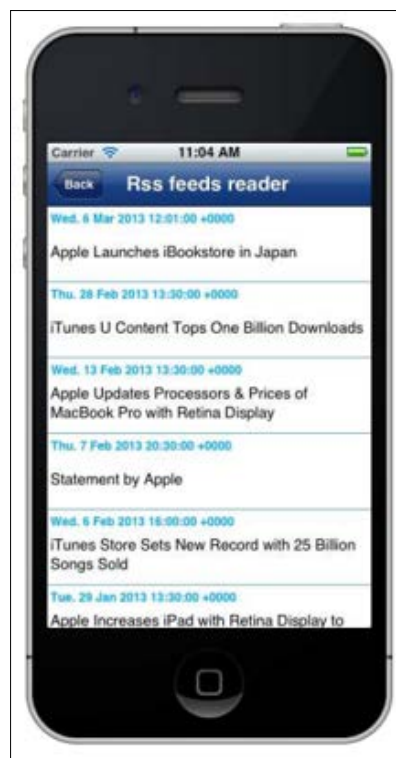
The above image shows a comparative analysis of cross-platform development approaches used for Android and iOS. It contains web, Hybrid, Generated and interpreted applications. From this, it can be observed that the market development factor is not applied to native applications. However, it is applied on the hybrid, interpreted and generated applications. The next factor is widespread technology. These technologies are applied to the

native, hybrid, and interpreted applications but not to generated applications. The next factor is related to hardware and data access. It is limited to all three types of applications including Native, Hybrid, and Interpreted but not to Generated applications. Another factor is the user interface and look. It shows that web and hybrid applications are simulated. On the other hand, the interpreted and generated applications are native. Lastly, the user-perceived performance level is high for the generated application, medium for hybrid and interpreted applications, and low for web applications [2].

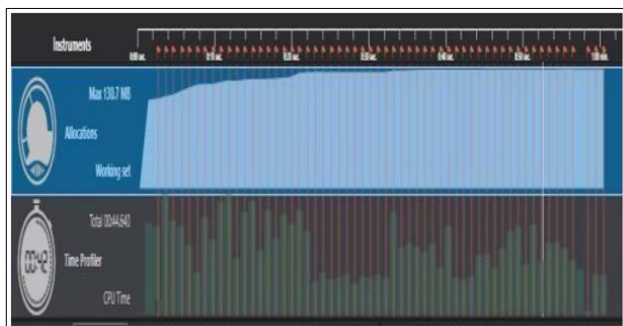


**Figure 3:** RSS Feed Reader for Android

The above image shows the information about the RSS feed reader for Android. Furthermore, for implementing an RSS feed reader on Android, the total source code is less than 150 lines and the client has tested these feed reader source codes on iOS too and its figure is given below that shows an RSS reader on iOS devices. Furthermore, every code line for the RSS feed is pure JavaScript without there is need to code anything using a native language for source code [1].



**Figure 4:** RSS Feeds Reader for iOS

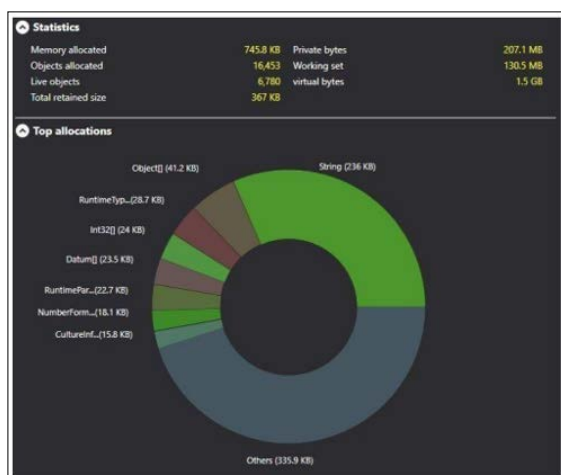


**Figure 5:** Memory Allocation Graph for Hybrid Application



**Figure 6:** Memory Allocation Graph for Native Application

The above two images show the information about the memory allocation graph for Hybrid and Native applications. This factor is corresponding to the performance of these applications. It shows that the performance factor of native applications is showing more accurate results compared with the hybrid applications. The graph shows that the native applications rarely use less memory compared with the hybrid applications. The use of memory is decreased with time in native applications but not in hybrid. Therefore, their performance is low compared with native applications [1].



**Figure 7:** Performance Statistics of Hybrid Application

The above graph shows the performance statistics of hybrid applications.

## Conclusion

Summing up all the discussion above, it is concluded that both native and hybrid applications will be upgraded with time and they contain some problems and advantages. However, the choice between hybrid and native application development depends on different factors that include budget constraints, project requirements, and performance expectations. On the other hand,

the native applications offer unparalleled performance and user experience. However, these applications come with high development costs and longer timeframes. Secondly, the hybrid applications were cost-effective and provided cross-platform development and compromise on performance and platform-specific user experience.

Due to the latest advancements, the future of mobile development is completely dynamic for both hybrid and native approaches. However, the native applications are highly dominant in performance-critical domains, and the hybrid frameworks are adapting and improving to emerging technologies. Under these facts, developers and businesses must be careful before accessing their specific needs and consider the evolving landscape for mobile technology while deciding between hybrid and native applications. It means that the main choice must align with the project's objective and long-term strategic goals [4-6].

## References

1. Choudhary RA, Tushar Mudi, Shilpi Sharma (2020) A Comparative Analysis of Native and Hybrid Mobile Application Development [http://www.junikhyatjournal.in/no\\_10\\_may\\_20/9.pdf](http://www.junikhyatjournal.in/no_10_may_20/9.pdf).
2. Sharjeel Moqrab Khan, Aftab ul Nabi, Tahir Hussain Bhanbhro (2022) Comparative Analysis between Flutter and React Native. International Journal of Artificial Intelligence & Mathematical Sciences 1: 16-29.
3. K Presa, Svensson Oskar (2021) React Native and native application development: A comparative study based on features & functionality when measuring performance in hybrid and native applications. <http://www.diva-portal.org/smash/get/diva2:1547719/FULLTEXT01.pdf>.
4. Ansong ED, Synaepa-Addision TQ (2019) A comparative study of user data security and privacy in native and cross-platform Android mobile banking applications. In 2019 International Conference on Cyber Security and Internet of Things (ICS IoT) <https://ieeexplore.ieee.org/document/9058363/authors#authors>.
5. Paulo RM de Andrade, Adriano B Albuquerque, Otávio F Frota, Robson V Silveira, Fátima A da Silva (2015) Cross-platform app: a comparative study. arXiv 7: 1-8.
6. Xanthopoulos S, Xinogalos S (2013) A comparative analysis of cross-platform development approaches for mobile applications. In Proceedings of the 6th Balkan Conference in Informatics, 2013 [https://www.researchgate.net/publication/258010031\\_A\\_Comparative\\_Analysis\\_of\\_Cross-platform\\_Development\\_Approaches\\_for\\_Mobile\\_Applications](https://www.researchgate.net/publication/258010031_A_Comparative_Analysis_of_Cross-platform_Development_Approaches_for_Mobile_Applications).

**Copyright:** ©2022 Abhishek Shukla. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.