JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

A Comparative Analysis of React Native and Apache Cordova for Android App Development

Shubham Pandey
Project Guide : Prof. Uravshi Rakholiya

Parul Institute of Technology Parul University

Vadodara Gujarat, India

Abstract: ***This research paper provides a comprehensive comparison between React Native and Apache Cordova frameworks for developing Android applications. React Native and Apache Cordova are two popular frameworks used by developers to build cross-platform mobile applications. The paper evaluates various aspects such as performance, development time, user experience, community support, and ecosystem to determine which framework offers the most suitable solution for Android app development. Through a thorough examination of the strengths and limitations of each framework, this paper aims to assist developers in making informed decisions when selecting the appropriate technology for their Android app projects.***

Keywords: Android Application; React Native, Apache Cordova; Cross-platform tools; Cross platform framework; Hybrid Mobile Application; Cross-platform mobile app development, React Native

I. INTRODUCTION

Mobile app development presents challenges in selecting the right technologies for multiple platforms, driving companies and developers towards cross-platform solutions to reach wider user bases. Cross-platform development simplifies processes, saving time and effort. Frameworks like React Native, Apache Cordova, Xamarin, and Flutter facilitate this by offering tools and libraries. React Native, an open-source framework by Facebook, enables high-quality app development for iOS and Android using a single codebase. Leveraging ReactJS and native components, it ensures seamless user experiences. Apache Cordova, also known as PhoneGap, facilitates hybrid app development using web technologies like HTML, CSS, and JavaScript. Its plugin architecture grants access to native device capabilities, promoting rich functionality. Cordova's unified development experience and extensive community support make it efficient and adaptable. This paper compares React Native and Apache Cordova, focusing on development flexibility, access to native APIs, performance, community support, and ease of use. Additionally, we conduct a pragmatic evaluation of these frameworks based on application file sizes, aiming to provide insights into their efficiency and suitability for cross-platform development.

1.1 Overview of React Native and Apache Cordova:

React Native is an open-source framework developed by Facebook, designed to build native-like mobile applications using JavaScript and React. It allows developers to write code once and deploy it across both iOS and Android platforms, while still offering the performance and user experience of native apps.

Apache Cordova, formerly known as PhoneGap, is another popular open-source framework used for cross-platform mobile app development. Developers have the capability to construct mobile applications utilizing web technologies like HTML, CSS, and JavaScript through it. Alos Cordova is an open-source framework. It provides a bridge between web content and native device features, enabling developers to access device APIs through JavaScript interfaces.

1.2. Performance Comparison:

Performance is a critical factor in determining the suitability of a framework for Android app development. React Native leverages native components, resulting in better performance compared to Apache Cordova, which relies on web views for rendering. However, the performance gap between the two frameworks has narrowed in recent years, with improvements in Cordova's rendering engine.

1.3 Development Time and Productivity:

React Native offers hot reloading, allowing developers to see the changes in real-time without recompiling the entire application. This feature enhances productivity and reduces development time significantly. On the other hand, Apache Cordova's development process may be slower due to the need for frequent recompilation and testing across multiple platforms.

1.4 User Experience:

Both React Native and Apache Cordova can deliver a native-like user experience, but React Native's use of native components provides a more seamless and responsive user interface. Cordova apps, while capable of achieving native-like behavior, may suffer from performance issues and inconsistencies in user experience across different devices.

1.5 Community Support and Ecosystem:

React Native boasts a large and active community, with extensive documentation, libraries, and tools available to developers. This robust ecosystem contributes to faster development and easier troubleshooting. Apache Cordova also has a strong community support, but it may not be as extensive as React Native's ecosystem.

1.5 Impact on Application File Sizes:

Research investigates the variability in file sizes of the same application developed using React Native and Apache Cordova. Results show that the application built using Apache Cordova has significantly smaller app download size and app size on device compared to the one built using React Native. Apache Cordova's optimization techniques contribute to creating smaller, more efficient applications, which can lead to faster download and installation, reduced storage requirements, improved performance, lower development and maintenance costs, and better user experience.

1.6 React Native vs Cordova: In-Depth Comparison

React Native and Cordova diverge primarily in their methodologies for app development. Cordova functions as a hybrid application framework, seamlessly encapsulating web technologies within a native container. Conversely, React Native facilitates the development of genuinely native applications by leveraging JavaScript and React components. Nevertheless, there are additional distinctions worth exploring between these two platforms. Let's delve into these further disparities.

1. Coding Method:

Cordova: "Write once, run anywhere" approach.

React Native: "Learn once, write anywhere" approach with components representing native components.

2. Documentation:

Cordova: Detailed documentation for smooth development experience.

React Native: Impressive support community working constantly to help developers succeed.

3. Compatibility:

Cordova: Better support for older versions of mobile systems.

React Native: Less support for older versions, some older devices may not run apps smoothly.

4. Debugging Capabilities:

Cordova: Relies on Safari's Web Inspector for iOS and Chrome Dev Tools for Android.

React Native: Offers convenient debugging options through Chrome Dev Tools.

5. Development Workflow:

Cordova: Customizable workflow, easy device integration.

React Native: Automatic updating of changes, ensuring smooth development.

6. Extensibility and Stability:

Cordova: Extensive control for creating and utilizing plugins, enhances code reusability.

React Native: Easy integration with various JavaScript modules, but may pose challenges in adapting code from web to mobile.

II. CONCLUSION

While both React Native and Apache Cordova offer viable solutions for cross-platform development, each has its strengths and weaknesses. React Native excels in providing a truly native experience with better runtime performance, making it a preferred choice for high-performance apps. On the other hand, Cordova offers ease of setup, compatibility with older systems, and a thriving plugin ecosystem, making it suitable for simpler apps or those requiring broad compatibility. The choice between the two frameworks ultimately depends on the specific requirements and priorities of the project.

III. ACKNOWLEDGMENT

I extend my heartfelt gratitude to our esteemed Project Guide, Prof. Urvashi Rakholiya, for her invaluable guidance and unwavering support throughout the duration of my project. Hiexpertise and insightful discussions have been instrumental in shaping our work and achieving our goals.

I would also like to express our sincere appreciation to our Head of Department, Prof. Sumitra Menaria, and our Project Coordinator, Prof. Urvashi Rakholiya, for their invaluable advice and guidance at every step of the way. Their encouragement and mentorship have been instrumental in navigating challenges and ensuring the success of our project.

Lastly, I am deeply thankful to our respected Principal, Dr. Swapnil Parikh, for providing us with the necessary resources and opportunities to bring our project to fruition. His continuous support and encouragement have been instrumental in our journey towards achieving excellence.

IX. References:

- Facebook. "React Native." Retrieved from https://reactnative.dev/
- Apache Cordova. "What is Apache Cordova?" Retrieved from https://cordova.apache.org/
- Joorabchi, Mona Erfani, Ali Mesbah, and Philippe Kruchten. "Real challenges in mobile app development." 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE, 2013.
- Martinez, Matias, and Sylvain Lecomte. "Towards the quality improvement of cross-platform mobile applications." 2017 IEEE/ACM 4th International Conference on Mobile Software Engineering and Systems (MOBILESoft). IEEE, 2017.
- Palmieri, Manuel, Inderjeet Singh, and Antonio Cicchetti. "Comparison of cross-platform mobile development tools." 2012 16th International Conference on Intelligence in Next Generation Networks. (pp. 179-186). IEEE, 2012.

- Masiello, Eric, and Jacob Friedmann. Mastering React Native. Packt Publishing Ltd, 2017.
- Shah, Kewal, Harsh Sinha, and Payal Mishra. "Analysis of cross-platform mobile app development tools." 2019 IEEE 5th International Conference for Convergence in Technology (I2CT). IEEE, 2019.
- Nawrocki, Piotr, et al. "A comparison of native and cross-platform frameworks for mobile applications." Computer 54.3 (2021): 18-27.