

WHY LEARN FLUTTER

Tursunbek Sadriddinovich Jalolov

Asia international university pffd .(PhD)

Annotation. In the modern mobile development industry, Flutter has emerged as one of the fastest-growing cross-platform frameworks. While native languages such as Swift, Kotlin, and Java, or alternative frameworks like React Native, Xamarin, and Ionic remain popular, Flutter offers a unique balance of speed, simplicity, and universality. This article explores the advantages of Flutter, compares it to other mobile programming languages and frameworks, and highlights why learning Flutter is strategically important for developers in today's global IT market.

Keywords: Flutter, Dart, Swift, Kotlin, Java, React Native, Xamarin, Ionic, cross-platform development, mobile applications.

Introduction. Mobile applications have become an inseparable part of everyday life. Businesses, governments, and individuals all rely on mobile solutions for communication, services, and productivity. This rapid digitalization raises an important question for developers: which programming language or framework should be chosen to create mobile apps efficiently?

Traditionally, iOS apps are developed in Swift and Android apps in Kotlin or Java. While these native approaches guarantee high performance, they require separate teams, separate codebases, and therefore more time and financial resources. The demand for faster, cheaper, and more scalable solutions has led to the rise of cross-platform frameworks. Among them, Google's Flutter stands out as a technology that allows developers to create high-quality applications for both Android and iOS using a single codebase.

Main Part

Flutter is based on the Dart programming language, which was also developed by Google. Dart is relatively easy to learn, especially for those who are already familiar with programming concepts such as object-oriented programming, asynchronous operations, and functional programming structures. By mastering Flutter, learners automatically gain competence in Dart, which further enriches their programming skills. One of the strongest arguments for learning Flutter is its cross-platform functionality. Traditionally, developers had to build separate applications for iOS and Android, which meant learning two different programming languages and frameworks such as Swift for iOS and Kotlin or Java for Android. This process was expensive, time-consuming, and resource-demanding. Flutter solves this problem by allowing developers to create high-performance applications for both platforms using a single codebase. This not only saves time and money, but also ensures consistency in design and performance. Another reason to learn Flutter is the speed of development. Thanks to features like "hot reload," developers can instantly see the results of their changes without restarting the entire application. This drastically shortens the development cycle, improves productivity, and makes

experimentation easier. For students and beginners, this is especially motivating, because they can observe their progress in real time and quickly understand the connection between code and user interface. In a learning context, this accelerates comprehension and fosters creativity.

The user interface capabilities of Flutter are another crucial aspect. Flutter uses its own rendering engine, which allows it to deliver highly customizable, smooth, and attractive designs. Developers are not restricted to platform-specific UI components, but can instead build their own unique widgets. This gives them greater freedom to create applications that stand out in the market. Businesses benefit from this because a visually appealing application can significantly enhance user engagement and customer satisfaction. Thus, Flutter is not just a technical tool, but also a means of artistic and creative expression.

Moreover, Flutter has a strong and active community that constantly develops new packages, libraries, and learning resources. Beginners can find tutorials, open-source projects, and community support easily, which reduces the difficulty of learning. The open-source nature of Flutter also ensures that it evolves continuously with contributions from developers worldwide. This community support is invaluable for anyone starting their journey with Flutter, as it provides not only technical guidance but also motivation and inspiration. In addition to these practical benefits, there are psychological and educational reasons for learning Flutter. For students, learning Flutter provides a sense of achievement because they can quickly build functional and beautiful applications, which boosts confidence and motivation to continue learning programming. For educators, Flutter represents a practical framework to introduce programming concepts, user interface design, and problem-solving skills. In business education, Flutter is highly relevant because it teaches students how to create cost-effective solutions that are aligned with industry needs. The professional advantages of learning Flutter are equally significant. Many companies around the world are adopting Flutter for their projects, which increases the demand for Flutter developers. By learning Flutter, professionals increase their employability and open opportunities for freelance work, startups, and corporate positions. For entrepreneurs, mastering Flutter provides the ability to quickly prototype and launch digital products without relying heavily on external developers. This accelerates innovation and empowers individuals to turn their ideas into reality.

Another important point is that Flutter is not limited to mobile development. With the introduction of Flutter for web and desktop, developers can now build applications that run on multiple platforms with the same codebase. This means that a single skill set allows one to create software for smartphones, tablets, browsers, and desktop computers. Such versatility is rare in the software development world, making Flutter an exceptional choice for learners who want to maximize the impact of their skills.

Flutter vs. Swift/Kotlin

Swift and Kotlin are native languages officially supported for iOS and Android development, respectively. They deliver excellent performance and seamless integration with their platforms. However, their limitation lies in exclusivity: Swift works only for iOS, while Kotlin is mainly for Android. This means two separate projects, two teams, and higher costs. Flutter solves this

problem by enabling developers to build apps for both platforms from one codebase, significantly reducing development time and expenses.

Flutter vs. Java

Java has been the backbone of Android development for many years. It is stable, has a massive ecosystem, and benefits from a large pool of experienced developers. However, Java often requires writing more boilerplate code, and building modern user interfaces can be more complex. Flutter, with its widget-based architecture, allows developers to create beautiful and responsive UIs quickly. Furthermore, Flutter's apps achieve near-native performance, making it a strong alternative to Java for Android projects.

Flutter vs. React Native

React Native, powered by JavaScript, is another popular cross-platform solution. It provides a unified codebase for Android and iOS, which makes it attractive to web developers transitioning to mobile. However, React Native relies on a "bridge" to communicate with native components, which may cause performance issues in complex apps. Flutter avoids this limitation by using its own rendering engine, ensuring smoother animations and faster performance. In addition, Flutter comes with a richer set of pre-built widgets compared to React Native, reducing the reliance on third-party libraries.

Flutter vs. Ionic

Ionic is built on web technologies such as HTML, CSS, and JavaScript. It is easy to learn and useful for simple apps, but its performance is limited because applications essentially run inside a web view. By contrast, Flutter produces apps that feel truly native thanks to its rendering engine. This makes Flutter more suitable for high-performance, visually demanding applications.

Comparative Summary

Performance: Flutter matches native languages (Swift, Kotlin) and outperforms React Native and Ionic. **Universality:** Unlike most frameworks, Flutter supports mobile, web, and desktop development with the same codebase. **Learning Curve:** Flutter uses Dart, a modern and beginner-friendly language, easier to pick up for developers familiar with JavaScript or Python. **Ecosystem & Community:** While React Native currently has a slightly larger ecosystem, Flutter's community is growing rapidly with strong support from Google. **Business Efficiency:** Flutter offers the most cost-effective approach by reducing time-to-market and resource requirements.

Conclusion

The mobile development landscape offers multiple options: Swift and Kotlin for native apps, Java for legacy Android, or cross-platform solutions like React Native, Xamarin, and Ionic. Yet Flutter stands out as the most balanced and efficient framework, combining high performance, universality, design flexibility, and ease of learning. For developers, learning Flutter is more than just acquiring a new skill—it is a strategic investment in their careers. It opens opportunities

across mobile, web, and desktop, ensures global competitiveness, and provides access to a rapidly expanding market. In today's fast-paced IT world, mastering Flutter is not only beneficial but essential for those who want to remain relevant and in demand.

References:

1. Jalolov, T. S. (2024). АЛГОРИТМЫ ПЛАНИРОВАНИЯ И ПРИНЯТИЯ РЕШЕНИЙ ДЛЯ РОБОТОТЕХНИКИ. Advanced methods of ensuring the quality of education: problems and solutions, 1(3), 73-79.
2. Jalolov, T. S. (2024). ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ КИБЕРБЕЗОПАСНОСТЬ В СИСТЕМАХ ПРИМЕНЯТЬ УГРОЗЫ. Advanced methods of ensuring the quality of education: problems and solutions, 1(3), 66-72.
3. Jalolov, T. S. (2024). AI YORDAMIDA QAYTA TIKLANUVCHI ENERGIYA MANBALARINI OPTIMALLASHTIRISH. Modern digital technologies in education: problems and prospects, 1(2), 72-77.
4. Jalolov, T. S. (2024). ATROF-MUHIT MONITORINGIDA SUN'IY INTELLEKT TEXNOLOGIYALARINING QO 'LLANILISHI. Modern digital technologies in education: problems and prospects, 1(2), 78-84.
5. Jalolov, T. S. (2024). MATNNI QAYTA ISHLASH ORQALI TIL O 'RGATISH ILOVALARINI RIVOJLANTIRISH. Modern digital technologies in education: problems and prospects, 1(2), 103-108.
6. Jalolov, T. S. (2024). OVOZLI KO 'MAKCHILARNING SAMARADORLIGINI OSHIRISH UCHUN CHUQUR O 'QITISH USULLARI. Modern digital technologies in education: problems and prospects, 1(2), 85-90.
7. Jalolov, T. S. (2024). SUN'IY INTELLEKTNI KIBERXAVFSIZLIK TIZIMLARIDA QO 'LLASH: TAHDIDLARNI ERTA ANIQLASH USULLARI. Modern digital technologies in education: problems and prospects, 1(2), 54-59.
8. Jalolov, T. S. (2024). KUCHLI VA ZAIF SUN'IY INTELLEKT MODELLARI: ULARNING TAQQOSLANISHI VA RIVOJLANISH ISTIQBOLLARI. Modern digital technologies in education: problems and prospects, 1(2), 91-96.
9. Jalolov, T. S. (2024). MASHINA O 'QITISH ALGORITMLARINI OPTIMALLASHTIRISH: SAMARADORLIK VA ANIQLIKNI OSHIRISH USULLARI. Modern digital technologies in education: problems and prospects, 1(2), 97-102.
10. Jalolov, T. S. (2024). SUN'IY INTELLEKT YORDAMIDA SOXTA MA'LUMOTLARNI ANIQLASH USULLARI. Modern digital technologies in education: problems and prospects, 1(2), 47-53.