

ТИПИ МОБІЛЬНИХ ЗАСТОСУНКІВ, ЇХ ОСОБЛИВОСТІ ТА ІНСТРУМЕНТИ ЇХ РЕАЛІЗАЦІЇ

Вінницький національний технічний університет

Анотація. У роботі досліджено типи мобільних застосунків та мови програмування від яких вони залежать. Розглянуто основні типи та їх вплив на процес розробки, витрати, час виконання та ефективність підтримки. Проаналізовано випадки використання, недоліки та переваги кожного з типів для оптимізації виконавчих процесів.

Ключові слова: нативна розробка, кроссплатформена розробка, гібридна розробка, вплив на процес розробки.

Abstract. The paper examines the types of mobile applications and the programming languages they depend on. It examines the main types and their impact on the development process, costs, execution time, and support efficiency. It analyzes the use cases, disadvantages, and advantages of each type for optimizing execution processes.

Keywords: native development, cross-platform development, hybrid development, impact on the development process.

Вступ

Мобільні пристрої стали невід'ємною частиною повсякденного життя людей, тому розробка мобільних додатків є надзвичайно важливою у сучасному світі. Саме мобільні додатки дозволяють нам легкий та швидкий доступ до неймовірно великої кількості функцій, такі як камера, GPS, доступ до інтернету, дзвінки, тощо. Також, мобільні додатки дозволяють мати доступ до сервісів різноманітних компаній та підприємств, і їх кількість постійно зростає, адже вони дозволяють охопити величезну аудиторію та є необхідністю для підтримки конкурентоспроможності на ринку.

Вибір типу розробки мобільного додатку є одним з найважливіших етапів у створенні успішного продукту, оскільки він визначає не лише технічні характеристики додатку, а й впливає на його функціональність, швидкість роботи, вартість розробки, а також досвід користувача. Існують різні підходи до розробки мобільних додатків, кожен з яких має свої переваги і недоліки: нативна, кроссплатформна та гібридна розробка. Кожен з цих методів має свої особливості, що зумовлюють їхній вибір залежно від конкретних вимог проекту. Саме тому важливо правильно оцінити, який підхід буде найбільш ефективним для досягнення поставлених цілей, мінімізації витрат та забезпечення високої якості кінцевого продукту.

Мета дослідження – визначити як кожен з методів впливає на процес розробки та за яких умов може використовуватись.

Нативний тип розробки застосунків

Нативний тип розробки мобільних додатків окремо розробляються для кожної платформи окремо. Вони можуть використовувати всі переваги функцій пристрою, такі як вібрація, камера, GPS, тощо. Вони можуть використовуватись без необхідності доступу до мобільної мережі, оскільки вони повністю завантажені на пристрій, а вже розроблені настанови дозволять полегшити створення інтерфейсу застосунку, та дозволить користувачам ознайомитись з ним швидше завдяки загальним характеристикам. Нативні застосунки також мають кращий контроль над орієнтацією, розміром та розширенням зображення та доступ до особливостей компонування, що дозволяють пришвидшити розробку.

Цей тип також обмежує гнучкість розробки, оскільки вони розробляється для специфічної платформи, вони вимагають окремого досвіду для кожної платформи, більшої витрати ресурсів на розробку та обслуговування, вартість якої доволі висока та становить 15-20% з коштів на розробку. Також такі застосунки вимагають більшої затрати ресурсів пристрою, оскільки їх можливість працювати без доступу до мережі Інтернет часто вимагає більшого розміру застосунку.

Swift є найкращою мовою розроблення нативних додатків для платформи IOS та дозволяє створити надійний, зрозумілий та швидкий у виконанні код, а Objective-C є альтернативою, що часто використовується для підтримки старших додатків та для інтеграції з сервісами на базі C, що також надає доступ до великої кількості бібліотек та рішень.

Java досі є популярною у використанні для розробки чи підтримки нативних додатків на платформі Android, завдяки своєму довгому існуванню та має велику екосистему фреймворків, бібліотек та інструментів розробки,

але, попри свої переваги, поступово програє конкуренцію Kotlin, який має коротший синтаксис, кращі якості для розробників та продуктивність на рівні Java.

Кросплатформний тип розробки застосунків

Кросплатформний тип розробки застосунків дозволяє створювати один мобільний застосунок, що має здатність плавно працювати на декількох операційних системах. Такий підхід дозволяє використовувати частину, або ж увесь розроблений код та розгорнути додаток на декількох платформах без необхідності кодування “з нуля” для кожної платформи, що дозволяє зекономити час та витрати на розробку, оновлення та підтримку, а існування додатку на декількох платформах дозволяє досягнути ширшої аудиторії.

Такий тип розробки теж має свої обмеження, часто пов’язані з продуктивністю. Процеси оптимізації розробки кросплатформних застосунків можуть привести до негативного досвіду користувачів через меншу продуктивність в порівнянні з нативними застосунками, особливо для ресурсоемних функцій, а обмежений доступ до вбудованих можливостей пристрою часто робить реалізацію необхідних функцій складнішою та вимагати специфічних API для їх роботи. Можливі обмеження в UI/UX також вимагатимуть компромісів під час розробки додатків для різних платформ, та робить перехід з використання додатку на різних платформах менш гладким.

C# у поєднанні з Xamarin чи .NET MAUI дозволяє розробникам добре структурований код із простим синтаксисом, дозволяє уникнути проблем з сумісністю, але вимагає коригування кодової бази для роботи певних функцій на Android та iOS.

Гібридний тип розробки застосунків

Гібридний тип розробки мобільних застосунків теж мусить бути встановленим на пристрій для його роботи як і будь-який інший застосунок. Такі додатки відрізняються тим, що вони мають елементи нативних застосунків, розроблених для певної платформи, з елементами веб застосунків та веб сайтів, що працюють немов вони не були встановлені, а просто були запущені через браузер за допомоги мережі Інтернет. Такі додатки розгортаються за допомоги нативного контейнера, який використовує мобільний об’єкт WebView. Це дозволяє відображати контент створений з використанням веб технологій під час роботи застосунку незалежно від платформи та скоротити час та витрати на розробку та підтримку, оскільки основна частина коду пишеться лише раз.

Обмеження гібридного типу розробки сильно обмежує можливості розробки інтерфейсу, та не дозволяє йому отримати нативного відчуття, а використання нативного контейнера не дозволяє додатку користуватися повним функціоналом пристрою. Крім того, комбінація обмежень кожної платформи вимагає від розробників використання необхідних плагінів, що ще більше ускладнює розробку проектів. Також гібридні додатки завжди залежатимуть від доступу до мережі Інтернет та будуть обмежені його швидкістю.

Саме через використання таких веб технологій, як HTML, CSS та JavaScript гібридний тип застосунків має найменшу з наведених типів необхідність в часі для розробки, оскільки команда може використовувати навички розробки веб застосунків, з якими вони вже знайомі.

Висновки

У процесі дослідження було визначено, що кожен з типів розробки має свої переваги та недоліки, тому кожен з них обирається командою в залежності до потреб замовника.

Таким чином, нативний метод варто використовувати у випадках, коли застосунок потребує доступу до повного функціоналу платформи чи пристрою, може працювати без доступу до мережі Інтернет та має кращий інтерфейс. Також варто пам’ятати, що такі додатки зможуть працювати на тій платформі, для якої вони розроблялися, а їх адаптація на іншу платформу буде мати велику ціну в ресурсах та часі.

Для декількох платформ найкраще обирати кросплатформний тип розробки, особливо якщо необхідна лише частина функціоналу платформи. Код застосунків створених таким методом не потрібно буде створювати повністю окремо для кожної платформи, але дизайн та продуктивність буде відрізнятися на кожній платформі, що може зруйнувати ілюзію плавного переходу.

Найменш затратним є гібридний метод розробки мобільних застосунків, завдяки відносній простоті їх створення, але вони сильно залежатимуть від швидкості мережі Інтернет, до якої підключений пристрій, а їх дизайн більше наближений до веб додатків.

Отже для вибору методу розробки мобільного методу необхідно знати наскільки важливою для додатку є продуктивність, мобільність та кількість ресурсів доступних для розробки.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ:

1. List of the Best Languages for Mobile app development. URL: <https://itechcraft.com/blog/best-programming-languages-for-mobile-app-development> (дата звернення: 21.03.2024).
2. Native app | Definition. URL: <https://uxcam.com/glossary/native-app> (дата звернення: 21.03.2024).
3. What is cross-platform mobile development? URL: <https://www.jetbrains.com/help/kotlin-multiplatform-dev/cross-platform-mobile-development.html> (дата звернення: 21.03.2024).
4. Hybrid Apps: The benefits, limitations & consequences for your Testing Phases. URL: <https://www2.stardust-testing.com/en/blog-en/hybrid-apps> (дата звернення: 21.03.2024).

Гавриш Євгеній Олегович – студент групи 4ПІ-24б, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, м. Вінниця, e-mail: eugeneedge17@gmail.com

Ліщинська Людмила Броніславівна – д-р техн. наук, професор, професор кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: llb@vntu.edu.ua

Havrysh Yevhenii Olegovich – student of group 4PI-21b, Faculty of Information Technologies and Computer Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: eugeneedge17@gmail.com

Lishchynska Lyudmyla Bronislavivna – Dr. Sc. (Eng.), Full Professor, Professor of Program Engineering, Vinnytsia National Technical University, Vinnytsia, e-mail: llb@vntu.edu.ua