УДК 004.413.5

РАССВЕТ ПРОГРЕССИВНЫХ ВЕБ-ПРИЛОЖЕНИЙ (PWA)

Шынтемір Әсел Айдарқызы
asel_aidarovna@bk.ru
Студент 4-го курса специальности «Радиотехника, электроника и телекоммуникации»
ЕНУ им. Л.Н.Гумилева, Нур-Султан,Казахстан
Научный руководитель – А. Наурызбаев

На протяжении многих лет наблюдается постоянный рост спроса на мобильное программное обеспечение из-за постоянного увеличения количества смартфонов. Мобильные разработчики могут свободно использовать различные архитектуры или стратегии разработки, включая нативное приложение, мобильное веб-приложение, гибридное приложение и новое прогрессивное веб-приложение (PWA). PWA, который сочетает в себе особенности нативной и веб-стратегий разработки, появился как лучшая альтернатива другим подходам разработки из-за дополнительных преимуществ, таких как автономная возможность, фоновая синхронизация и так далее, несмотря на некоторые опасения, которые были подняты в отношении эффективности PWA. Таким образом, эта исследовательская работа направлена на проведение сравнительного исследования существующих архитектур мобильной разработки с использованием метода Систематического обзора литературы (SLR), проведение сравнения функций на нативной, гибридной и PWA-архитектуре и, наконец, аргументацию архитектуры разработки PWA на

основе сравнений. Сравнение поможет исследователям и разработчикам понять концепцию PWA, тем самым мотивируя их принять эту стратегию для дальнейшего развития.

PWA-это развивающаяся технология, которая постепенно приобретает академическое участие с точки зрения исследований.Команда разработчиков мобильного программного обеспечения или организации могут принять одну или несколько существующих стратегий разработки, начиная от нативных приложений и заканчивая мобильными веб-приложениями, гибридными приложениями и теперь PWA. Нативная стратегия разработки приложений оказалась первой из существующих, она состоит из двоичных исполняемых файлов, которые непосредственно загружаются и хранятся на мобильном устройстве пользователя . Приложения, разработанные с использованием этой архитектуры, зависят от платформы и распространяются исключительно через специальный магазин приложений (Google Play Store, Apple App Store, BlackBerry App World) в зависимости от платформы, адаптированной поставщиками мобильных устройств. Определил высокое время разработки, высокие затраты на тестирование и техническое обслуживание как основную проблему нативного приложения, назвал это проблемой мобильной фрагментации, которая подразумевает, что код, написанный для одной мобильной платформы (например, java-коды для Android-приложения), не может быть использован для другой платформы, такой как приложение Apple iOS, написанное на Objective-C. в попытке преодолеть проблемы нативного приложения, где каждая платформа имеет свой собственный набор разработки программного обеспечения (SDK) с различными возможностями разработки, было разработано несколько кросс-платформенных архитектур, которые позволяют развертывать мобильные решения с использованием одного SDK.

Эти подходы идентифицированы как веб-подход, который используется при разработке мобильных приложений с использованием веб-технологий (HTML, CSS и JavaScript), размещенных на удаленном сервере, что делает его независимым от платформы, поскольку доступ к оптимизированному для мобильных устройств веб-сайту/приложению осуществляется через браузерное приложение, такое как Chrome, Firefox или Safari, которое должно быть предварительно установлено на мобильных устройствах пользователя . Основная проблема этого подхода заключается в том, что доступ к приложениям осуществляется только через Единый локатор ресурсов (URL) с использованием надежного и постоянного подключения к Интернету, что означает, что приложения не могут быть загружены через различные магазины приложений. Гибридный подход в соответствии с пытался использовать преимущества нативной и веб-архитектуры. При гибридном подходе мобильные решения разрабатываются с использованием веб-технологий, но визуализируются внутри собственных приложений и распространяются через различные магазины приложений. Другими обсуждаемыми подходами были интерпретируемый подход, который использует общий язык программирования, такой как JavaScript, для написания кода, который, в свою очередь, генерирует эквивалентность для нативного компонента для каждой платформы, кросс-компиляционный подход, который позволяет разработчикам писать коды с использованием любого общего языка программирования, которые затем преобразуются кросс-компиляторами в конкретный нативный код.

Для преодоления проблем, связанных с различными подходами (архитектурами) мобильной разработки, выявленными вышеприведенными исследователями, был разработан еще один подход к разработке, известный как Прогрессивное веб-приложение (PWA). Biørn-Hansen и его команда провели измерение-сравнение размера установки, времени запуска и времени от рендеринга панели инструментов app-icon tap среди гибридных, интерпретируемых и PWA мобильных подходов разработки. Результат показал, что PWA имеет наименьший размер установки, а также наименьшее время запуска.

Нативные приложения (Native Applications)

Нативные приложения-это приложения, разработанные с использованием инструментов и языков программирования, предназначенных для определенной мобильной платформы. Нативные приложения зависят от платформы, поэтому программисты должны

соответствовать конкретным языкам и инструментам, необходимым для успешной разработки приложения. Основным недостатком такого подхода к разработке является фрагментация мобильной платформы, что означает, что для того, чтобы фирма–разработчик могла охватить больше аудитории на разных платформах, должна быть "повторная разработка" одного и того же приложения на разных технологиях и инструментах, специфичных для каждой желаемой платформы. Это приводит к увеличению времени разработки, стоимости разработки, усилий, затрат на техническое обслуживание и низкой переносимости. На рисунке 1 схематично показан подход нативной мобильной разработки.

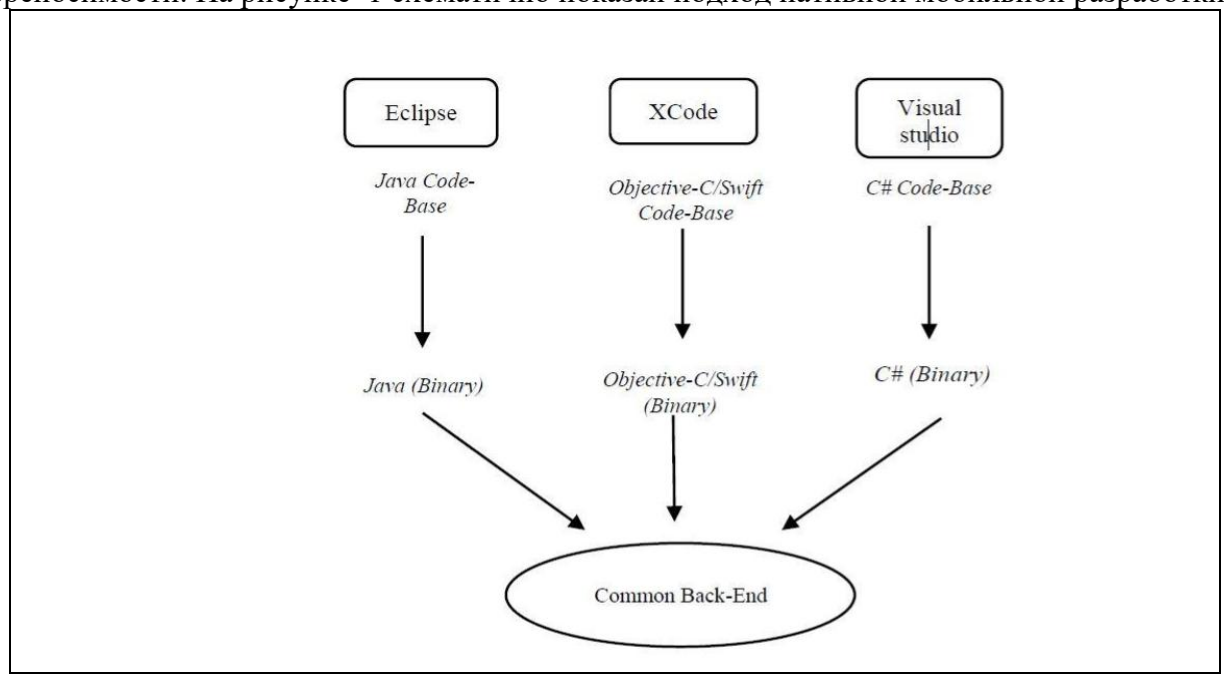

Рисунок 1. Подход к разработке Мобильных Нативных Приложений

Гибридный подход (Hybrid Approach)

Этот подход пытается использовать преимущества как нативного, так и веб-подхода, тем самым преодолевая некоторые ограничения, связанные с обоими подходами. Приложения, разработанные с использованием гибридного подхода, используют браузерный движок в мобильном устройстве и встраивают HTML-контент в собственный веб-контейнер (например, WebView для Android, UIWebView для iOS) . Предоставление некоторых мобильных гибридных фреймворков разработки, таких как Cordova, Ionic, PhoneGap, MoSync, обеспечивает собственную оболочку, содержащую веб-коды, а также универсальный JavaScript API, который служит мостом запроса сервиса от веб-кода к соответствующему API платформы .Гибрид использует преимущества как веб-приложения, так и нативного приложения. На рисунке 3 показано схематическое представление гибридного подхода.

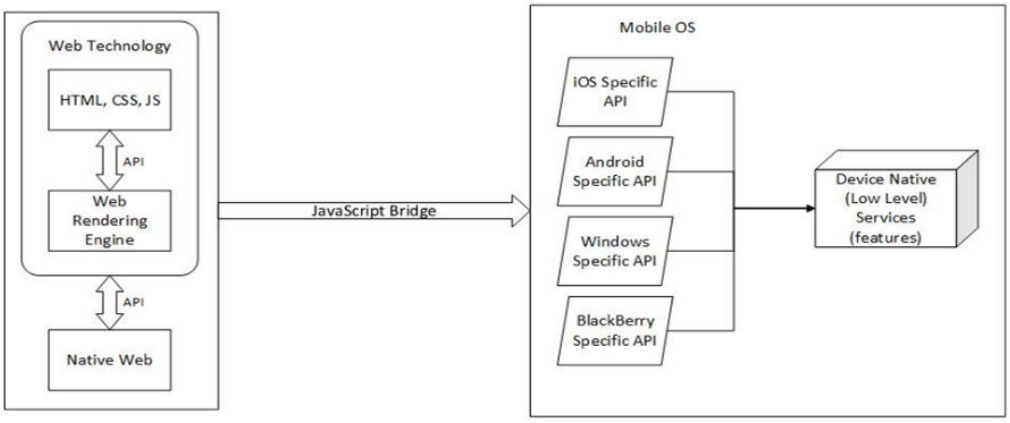

Рисунок 3. Схематическое представление гибридного подхода к разработке мобильных устройств

Прогрессивное веб-приложение (PWA)

PWA-это мобильный подход к разработке, направленный на преодоление проблем или слабых сторон предыдущих подходов. Принятие этого подхода создает особый вид веб-приложений, которые не требуют установки перед использованием и обслуживаются с удаленного сервера по защищенному протоколу передачи гипертекста (HTTPS) в отличие от обычных мобильных веб-приложений, которые могут обслуживаться с помощью HTTP . Пользователь PWA получает нативное приложение, подобное опыту, продвигая PWA в мобильное приложение верхнего уровня с полноэкранной поддержкой (без браузера) после принятия решения об установке PWA на устройство пользователя .PWA основан на концепции единого приложения для всех платформ , как и гибридный подход. Однако он обладает отличными возможностями, такими как мгновенная загрузка, push-уведомление даже в автономном режиме. На рисунке 5 схематично показан подход к разработке PWA.

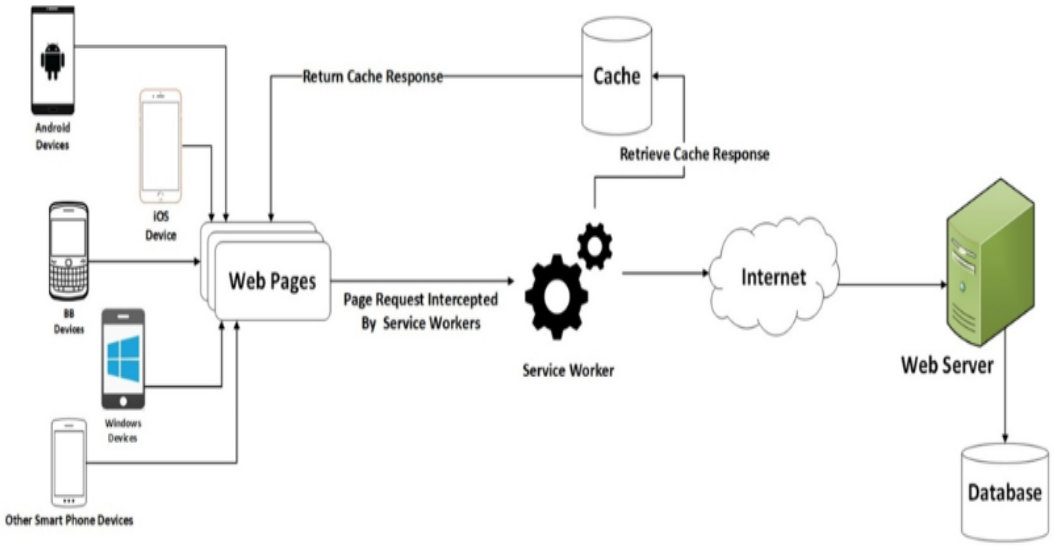

Рисунок 5. Архитектура подхода к разработке PWA

Компания googleband составила список рассматриваемых функций, которые являются базовыми требованиями для PWA, как указано ниже.

1.Автономные возможности: PWAS имеют возможность работать в значительной степени, даже если устройство находится в автономном режиме (режим полета или вне зоны действия сети).

2.Push-уведомление: PWAS имеют возможность отображать повторно привлекательные уведомления, как определено в push API.

3.Добавить на главный экран: Возможность установки веб-приложения на устройство пользователя по желанию.

4.Фоновая синхронизация: Возможность синхронизации данных в фоновом режиме.

5.Оценка:возможность оценить доступное хранилище, которое использует приложение, а также узнать объем оставшегося хранилища.

6.Веб-ресурс: Возможность использовать собственный виджет общего доступа, принадлежащий операционной системе (ОС), как указано API веб-ресурса.

7.Кроссбраузерное использование: возможность работы с основными браузерами.

8.Уникальная идентичность страницы: каждая страница имеет уникальный URL-адрес, который делает ее связываемой с другими страницами.

9.Платежный запрос: Возможность использовать API web payment request для работы в качестве посредника между продавцами и пользователями.

Выявленные особенности сделали PWA особым видом мобильного веб-приложения выделил четыре области, в которых PWA нацелен на улучшение общего веб-опыта, как указано ниже:

1.Конверсия: PWA основаны на прогрессивной стратегии улучшения, в которой функции нижнего уровня первоначально кэшируются, после чего расширенные функции (в зависимости от браузера) постепенно вводятся в действие.

2.Надежность: С помощью сервисных работников, PWAs могут быть загружены мгновенно с низким или без сетевого подключения –зависимости от сетей устраняются.

3.Производительность: Существует постоянный фоновый процесс сервисных работников, чтобы обеспечить мгновенный и надежный опыт для пользователей.

4.Вовлеченность: Привлечение пользователей было сделано легко с помощью PWA, поскольку он поддерживает push-уведомление в облаке.

Обещания, предлагаемые PWAs, нельзя ни недооценивать, ни сравнивать с существующими (традиционными) стратегиями мобильного развития. Фирмы –разработчики стремятся сократить время разработки, время тестирования и затраты, а также общие затраты на техническое обслуживание, что относительно невозможно при внедрении нативной и гибридной архитектуры разработки. Подход к разработке мобильных веб - сайтов полностью устранил проблему фрагментации мобильных приложений, которая подразумевает, что мобильное приложение теперь может работать на любой мобильной платформе с помощью браузера и не нуждается в повторной разработке. PWA полностью привнесла новое измерение с помощью service worker, app shell и других компонентов, которые облегчили автономную загрузку, фоновую синхронизацию, push-уведомление мобильных приложений, тем самым сделав веб-приложения похожими на нативные и гибридные приложения. Это исследование дает рекомендацию PWA разработчикам мобильных приложений на основе сравнения и анализа функций. Однако дальнейшие эксперименты по подходу к разработке мобильных устройств могут быть проведены с точки зрения управления памятью и эффективности на смартфонах, чтобы еще больше подтвердить утверждения этой работы.

Список использованных источников

1. Bakaus P. What Are Progressive Web AMPs? [Электронный ресурс] // 2016. URL: https://www.smashingmagazine.com/2016/12/progressive-webamps/ .
2. Green I. From AMP to PWA [Электронныйресурс] // 2016. URL: h
3. B. Y. K. Taylor and L. Silver, “Владение смартфонами быстро растет по всему миру, но не всегда одинаково”, Pew Res. Цент., нет. Февраль 2019 г.

4. A. Russell, “Progressive Web Apps: Escaping Tabs Without Losing Our Soul", Unrequently Noted, 2015. [Онлайн]. Доступно: https://infrequently.org/2015/06/progressive-apps-escaping-tabs-without-losing-our-soul/.
5. Мобильные приложения. Анализ защищенности мобильных приложений [Электронный ресурс] // 2016. URL: https://dsec.ru/services/securityanalysis/mobile-applications.
6. Мэтью МакДональд «Создание Web-сайтов. Основное руководство» 2010 г. Издательство: Эксмо, - 768 с.