УДК 004.03

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ И СРЕДСТВ РАЗРАБОТКИ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

А.И. СТАТУТ
(Представлено: канд. физ.-мат. наук, доц. О.В. ГОЛУБЕВА)

Рассматриваются методы разработки мобильных приложений, такие как разработка нативных приложений и гибридных. Также в статье произведен анализ преимуществ и недостатков разработки мобильных приложений с помощью различных типов программных библиотек.

Существует множество различных библиотек и других средств и инструментов разработки мобильных приложений. Каждый из них имеет свою особенность и предоставляет уникальные возможности, поэтому в данном исследовании поставлена задача – рассмотреть те инструменты, которые являются наиболее подходящими для реализации большинства мобильных приложений.

Операционные системы

Однако перед тем как говорить о видах мобильных приложений, стоит рассмотреть мобильные операционные системы (платформы).

На сегодняшний момент, по официальным данным, среди мобильных операционных систем в мире (и в частности в Беларуси) лидирует Android и iOS [1].

С учетом этих данных можно обосновать выбор операционных систем для разработки мобильного приложения, так как чем больше людей используют ту или иную платформу, тем выше вероятность того, что разрабатываемое приложение станет распространенным. Это утверждение не означает, что если разработано приложение под Android, то его сразу же загрузят тысячи людей, однако, если же приложение ориентированно на людей, проживающих в Республике Беларусь, и под операционную систему Windows Phone, то оно определенно обречено на провал, так как пользователей данной системы в нашей стране крайне мало.

Таким образом, при разработке приложения нужно учитывать предпочтения пользователей, на которых вы ориентируетесь. Для разработки мобильных приложений следует использовать наиболее распространенные операционные системы, какими в Республике Беларусь являются Android и iOS.

Типы мобильных приложений

Существуют два типа мобильных приложений: нативные и гибридные[2].

Нативные приложения разрабатываются инструментами, предоставленными Apple для iOS, Google для Android, Microsoft для Windows Phone и т.д., и стандартным языком для выбранной платформы, используется API, которое предоставляет данная платформа; интерфейс таких приложений органичен и соблюдает стиль платформы.

Гибридные приложения чаще всего разрабатываются с помощью JavaScript, HTML5, CSS3. Инструменты, используемые для их написания, преобразуют веб-представление приложения в нативное, но в случае, если ведется разработка сложного приложения, уровни абстракции, используемые при преобразовании, чаще всего не позволяют использовать большинство, свойственных нативным приложениям, функциональных особенностей. Скорость работы таких приложений намного ниже нативных, а также они нуждаются в постоянном и непрерывном интернет-соединении. Перед тем как заняться разработкой, следует решить, для каких мобильных платформ будет реализовано приложение, сколько денежных средств будет вложено в проект.

Лучшим решением будет написание нативного приложения. Это обусловлено тем, что нативные приложения работают намного быстрее, а их интерфейс более стилизован. Но в этом случае затраты на разработчиков будут намного выше, так как потребуется команда специалистов с набором конкретных навыков. Если же требуется написать приложение, ориентированное на несколько платформ с наименьшими затратами и с удовлетворительным качеством, стоит задуматься о разработке гибридного приложения.

Анализ гибридных платформ

Если требуется разработать гибридное приложения, то, учитывая все недостатки данного вида приложений, необходимо отобрать несколько наиболее распространенных средств разработки и провести их сравнительный анализ. Рассмотрим следующие средства разработки:

1. IONIC.
2. AppceleratorTitanium.
3. ReactNative.
4. Cordova.

IONIC – это средство разработки приложений с открытым кодом для разработки мобильных гибридных приложений. Построен на основе AngularJS и Apache Cordova и обеспечивает разработку, используя такие веб-технологии как CSS, HTML5 и JavaScript. Приложения, созданные с помощью этих технологий распространяются через магазины приложений (Google Play или App Store и т.д.) [3].

Кроме функций, унаследованных от Apache, Cordova также предоставляет следующие возможности:

1. Выбор шаблона приложения.
2. Сборка и запуск в эмуляторе, на реальном устройстве, в браузере.
3. Генерация иконок и других элементов.

Поддерживаемые платформы: Android, iOS, Windows, BlackBerry.

Appcelerator Titanium – это фреймворк с открытым кодом, который позволяет создавать гибридные мобильные приложения с помощью JavaScript, разработанный Appcelerator.

Архитектурой этой библиотеки является кроссплатформенное API для доступа к нативным компонентам пользовательского интерфейса таким как: панели навигации, меню, диалоговым окнам, файловой система, сети, геолокации, акселерометр и карты. Весь исходный код приложения интерпретируется с использование механизма JavaScript. Загрузка программы занимает больше времени, чем нативная программа, так как интерпретатор и все необходимые библиотеки должны быть загружены до интерпретации исходного кода. Некоторые разработчики сообщают: хотя работа с титаном дает быстрые результаты, что хорошо подходит при создании прототипов, однако есть вопросы, связанные с различиями в поведении с кроссплатформенным API, стабильностью и управлением памяти, что заставляло их переписывать свои приложения с помощью нативного кода.

Поддерживаемые платформы: Android, iOS, Windows, BlackBerry.

React Native – это платформа для создания гибридных приложений на основе JavaScript библиотеки React [4]. В отличие от других подобных платформ, которые запускают JavaScript приложения в веб-браузере, React Native компилирует в машинный код для соответствующей платформы. Это означает, что никакого снижения производительности приложения, созданного с помощью него не будет и приложение будет работать также как, как и приложения построенные с помощью нативных инструментов. Однако, кроме очевидных плюсов, платформа имеет множество недостатков: старые версии платформы не поддерживаются новыми, нет хорошей документации, всю необходимую информацию придется искать на форумах, трудоемкий процесс отладки приложений. Их можно объяснить тем, что платформа вышло относительно недавно и очень стремительно развивается.

Поддерживаемые платформы: Android, iOS.

Apache Cordova – одна из наиболее популярных платформ для разработки мобильных приложений. Она позволяет создавать гибридные мобильные приложения для мобильных устройств с помощью CSS3, HTML5, JavaScript. Приложения выполняются в обертках, которые являются платформо-ориентированными и опираются на стандартные API привязки для доступа к возможностям каждого из устройств, таких как датчики, данные, состояния сети и т.д. [5].

Поддерживаемые платформы: AppleiOS, Bada, BlackBerry, FirefoxOS, GoogleAndroid, LGwebOS, MicrosoftWindowsPhone, Tizen (SDK 2.x) и UbuntuTouch.

Если сравнить приведенные выше платформы, то среди них мы можем выделить IONIC и React Native, так как они являются наиболее быстрыми и продвинутыми по сравнению с Apache Cordova и Appcelerator Titanium. Ниже рассмотрим их сравнительный анализ на основе простейшего приложения с картой.

Сравнение платформы IONIC и ReactNative

Рассмотрим сравнение простейшего приложения, содержащего карту, разработанного на IONIC и React Native. Сравнение будет производиться по следующим критериям: сторонние модули, размер приложения, использование памяти.

1. *Сторонние модули*

Разрабатывая на React Native, мы можем использовать сторонние модули. В данном случае будем использовать нативное представление Google карт. В IONIC мы также можем использовать сторонние плагины, однако в случае с картами мы должны использовать WebView для их отображения. Недостатком данного подхода является более долгая загрузка и использование большего объема памяти. В нативном представлении карты мы можем использовать трехмерное отображение, масштабирование и компас.

2. *Размер приложения*

IONIC использует уже существующее WebView для отображения приложения, поэтому его размер достаточно небольшой. В то же время React Native создает мост между JavaScript и нативными компонентами, что сказывается на размере приложения. Одно и то же приложение в IONIC имеет размер всего 2.88 MB, в то время как в React Native – 8.22 MB. Разница довольно велика, хотя файлы приложения React Native можно сжать с помощью ProGuard. Однако в данном случае это сжатия не было выполнено.

3. *Использование памяти*

Количество памяти, которое использует приложение, очень важно, особенно для медленных устройств на Android. Поэтому там необходимо сохранить низкий уровень использования памяти так, чтобы приложение могло бесперебойно работать на устройствах. Если мы запустим команду adb shell dumpsys meminfo, то получим следующий результат: Ionic Framework использует ~ 84,714 kB памяти, в то время как React Native – ~ 58,585 kB. Таким образом, приложение работает лучше на React Native.

Подводя итог можно сказать, что наилучшим средством разработки гибридного мобильного приложения является React Native.

Заключение

Подводя итог, можно сказать, что выбор разрабатывать нативное либо гибридное мобильное приложение необходимо тщательно проанализировать, так как у каждого подхода есть свои преимущества и недостатки.

Если необходимо разработать приложение в сжатые сроки и на несколько операционных систем одновременно, но, возможно, в ущерб производительности, то лучше выбрать гибридные платформы.

Если проект имеет достаточный бюджет, приложение будет иметь сложные элементы интерфейса и важна производительность, то выбор за нативным методом.

Если сравнивать гибридные платформы, ReactNative является одним из лучших решений как компромисс между другими гибридными платформами и нативным способом разработки.

ЛИТЕРАТУРА

1. Восемь факторов, влияющих на стоимость разработки мобильного приложения [Электронный ресурс]. – Режим доступа: https://link.springer.com/content/pdf/10.1007%2Fs13174-010-0007-6.pdf. – Дата доступа: 10.09.2017.
2. Мобильные приложения: нативные vs html5 vs гибридные [Электронный ресурс]. – Режим доступа: http://www.skywell.com.ua/blog/mobilnyie-prilozheniya-nativnyie-vs-html5-vs-gibridnyie/. – Дата доступа: 07.08.2017.
3. Ionic framework. Обзор экосистемы [Электронный ресурс]. – Режим доступа: https://habrahabr.ru/company/simpleweek/blog/254681/. – Дата доступа: 07.08.2017.
4. React Native [Электронный ресурс]. – Режим доступа: https://facebook.github.io/react-native/. – Дата доступа: 07.09.2017.
5. Введение в Apache Cordova [Электронный ресурс]. – Режим доступа: https://cordova.apache.org/docs/ru/latest/guide/overview/. – Дата доступа: 17.09.2017.