электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.9

Способы кроссплатформенной разработки мобильных приложений

***Ермаков О.Ю.**, студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»

Научный руководитель: Гапанюк Ю.Е., к.т.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Системы обработки информации и управления»
gapyu@bmstu.ru

Введение

По всем признакам, мы стоим на пороге мобильной интернет-революции, которая произойдет благодаря стремительному распространению мобильного интернета (по данным авторов статьи [1] - порядка 18,4 млн человек в России) и разработке IT-приложений, установленных на новых, интуитивно понятных пользовательских устройствах (iPad, iPhone, Samsung Galaxy, iStethoscope и т.д.). Причем собственное мобильное приложение становится хорошим тоном, визиткой, коммерческим инструментом и дополнительным источником дохода.

Поэтому разработка нового поколения мобильных приложений, использующих web-ресурсы и современные IT-технологии, становится популярной и потенциально выгодной.

Данная статья ориентирована на архитекторов, разработчиков и специалистов по информационным технологиям, которые проектируют, создают или используют мобильные приложения и сервисы, для разных операционных систем.

Краткий анализ существующих мобильных приложений

На данный момент у разработчиков есть два основных варианта создания мобильных приложений:

1. Нативные разработки - создание родного приложения для каждой поддерживаемой платформы. Это, безусловно, обеспечивает максимальное удобство для пользователя и производительность на каждой платформе, в то же время, открывая доступ ко всем встроенным аппаратным средствам и магазинам приложений. Однако недостаток этого подхода в том, что он может потребовать гораздо больших затрат на создание и

сопровождение, так как понадобятся раздельные кодовые базы для каждой платформы, которую необходимо поддерживать. Кроме того, каждая новая версия приложения потребует передавать его во все магазины приложения заново.

2. Кроссплатформенные разработки:

   Под кроссплатформенным мобильным приложением будем понимать такое мобильное приложение, которое работает более чем в одной операционной системе (или аппаратной платформе).

   - Создание мобильного веб-приложения. Это самый простой и дешевый вариант для разработки, выпуска и обновления на всех платформах, а также доступа к базе данных сервера, но удобство в использовании может пострадать из-за отсутствия доступа к встроенным аппаратным средствам.
   - Гибридные мобильные приложения (по мнению автора [2] - альтернатива(?) кроссплатформенным приложениям). Частично оптимизируются для разных платформ, на уровне отдельных модулей. Это позволяет существенно сэкономить на разработке, однако производительность гибридных приложений может быть хуже.

Важно отметить, что ни один из этих подходов не является принципиально лучшим других - все они имеют свои сильные и слабые стороны. Глубокий анализ соотношения "цены/выгоды" по каждому варианту поможет определить, какой из них будет лучше для пользователей конкретного бизнеса.[3]. При принятии решения важно учитывать удобство в использовании, затраты на разработку и последующие расходы на сопровождение, а также более тонкие факторы вроде маркетинга и признания среди пользователей.

Для многих бизнес - сценариев предпочтительнее вариант с гибридным мобильным приложением, так как дополнительные усилия и расходы на создание уникальных родных приложений для каждой мобильной платформы могут перевесить все выгоды для бизнеса. Мобильные приложения для бизнеса никуда уже не денутся и будут лишь набирать все большую значимость по мере переноса вычислений с традиционных настольных компьютеров на разнообразные мобильные устройства. [3].

Кроссплатформенные мобильные приложения

Сравнительные характеристики нескольких продуктов для создания кроссплатформенных мобильных приложений, таких как Appcelerator Titanium, Kony Platform, Adobe PhoneGap, IBM Worklight, Telerik Platform, Verivo Akula, Xamarin, представлены в [4], [5].

Краткие обзоры популярных типов гибридных приложений приведены в [6], [7], и знакомят с распространенными классами гибридных приложений, с инструментами кроссплатформенных разработок, которые используются для проектирования и работы гибридных приложений.

К сожалению, в данные обзоры не успели войти новые (2014 года) технологии для создания гибридных приложений, предложенные компанией Microsoft в сотрудничестве с компаниями Xamarin и Apache Software Foundation.

Итак, в последних версиях Visual Studio предложены следующие технологи для создания гибридных приложений:

- Нативная платформа и технологии (**Xamarin**):
  - Полный доступ к возможностям платформы;
  - Богатые возможности создания UI;
  - Разное поведение на разных устройствах.
- Гибридные приложения и веб технологии (**Apache Cordova**):
  - Единый код;
  - Легко поддерживается;
  - Ограничения по интеграции с устройствами и производительности.

Рассмотрим каждую из представленных технологий.

Технология Xamarin

Xamarin — это фреймворк для кроссплатформенной разработки мобильных приложений (iOS, Android, Windows Phone) с использованием языка C#. Фреймворк состоит из нескольких основных частей [8]:

- Xamarin.IOS — библиотека классов для C#, предоставляющая разработчику доступ к iOS SDK;
- Xamarin.Android — библиотека классов для C#, предоставляющая разработчику доступ к Android SDK;
- Компиляторы для iOS и Android;
- IDE Xamarin Studio;
- Плагин для Visual Studio.

Для каждой из платформы потребуется реализовать собственный слой UI, т.е код, который отвечает за внешний вид приложения. Это цена за возможность использования нативных механизмов работы с UI. Если разбивать приложение на слои, то получается схема, представленная на рисунке 1:

- Data Layer (Хранилище данных, DL) – Хранилище данных, например, база SqlLite или xml-файлы;
- Data Access Layer (Обертка над хранилищем данных, DAL) – Обертка над хранилищем для осуществления CRUD-операций;
- Business Layer (Бизнес-логика приложения BL) – Слой, содержащий бизнес-логику приложения;
- Service Access Layer (Взаимодействие с удаленными сервисами, SAL) – Слой, отвечающий за взаимодействие с удаленными сервисами (Rest, Json, WCF);
- Application Layer (Платформозависимый код, AL) – Слой, содержащий платформозависимый код, другими словами, это код, который зависит от библиотек monotouch.dll или monodroid.dll;
- User Interface Layer (Пользовательский интерфейс, UI) – Слой пользовательского интерфейса.

Рис.1. Структурная схема Xamarin приложения

Кроссплатформенными являются все слои, расположенные выше Application Layer. Доля переносимого кода достаточно сильно зависит от самого приложения. Инженеры Xamarin это понимают, поэтому стремятся к увеличению этой доли. В качестве достижений в решении этой проблемы можно рассматривать библиотеку Xamarin.Mobile. Она предоставляет единый для различных платформ API для работы с камерой,

контактами и гео-локацией. Но использование этой библиотеки никак не ограничивает в применении платформозависимого API, например, с помощью механизма делегатов.

Технология Apache Cordova

Apache Cordova — технология для создания кроссплатформенных мобильных приложений, использующих HTML5 и JavaScript.

Cordova предоставляет среду для хостинга вашего контента HTML5/JavaScript внутри тонкой «родной» оболочки (рис.2). Для ОС на каждой платформе смартфонов она использует родной элемент управления «браузер» для рендеринга контента вашего приложения, причем ресурсы приложения упаковываются в дистрибутив. В случае Windows Phone ваши HTML5-ресурсы упаковываются в XAP-файл и загружаются в изолированное хранилище, когда запускается ваше Cordova-приложение. В период выполнения элемент управления WebBrowser визуализирует контент и исполняет ваш JavaScript-код.

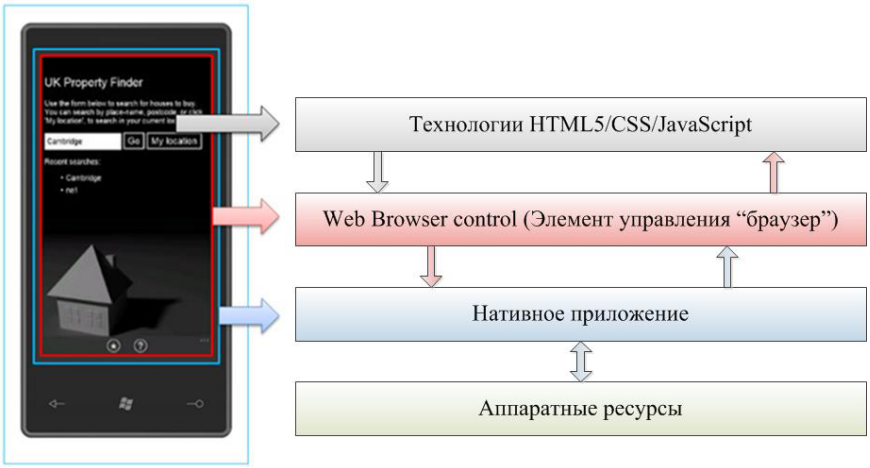

Рис. 2. Функциональная схема Apache Cordova приложения

Cordova также предоставляет набор стандартных API для доступа к функциям, общим для всех смартфонов. К некоторым из этих функций относятся:

- события жизненного цикла приложения;
- хранилище (локальное хранилище и базы данных HTML5);
- контакты;
- камера;
- геопозиционирование;
- акселерометр.

Каждая из этих функций предоставляется как JavaScript API, который используется из вашего кода на JavaScript. Cordova берет на себя всю черновую работу, связанную с

предоставлением необходимой родной реализации, и тем самым гарантирует, что вы будете иметь дело с одинаковыми JavaScript API независимо от ОС смартфона, на котором выполняется ваш код (рис. 3).

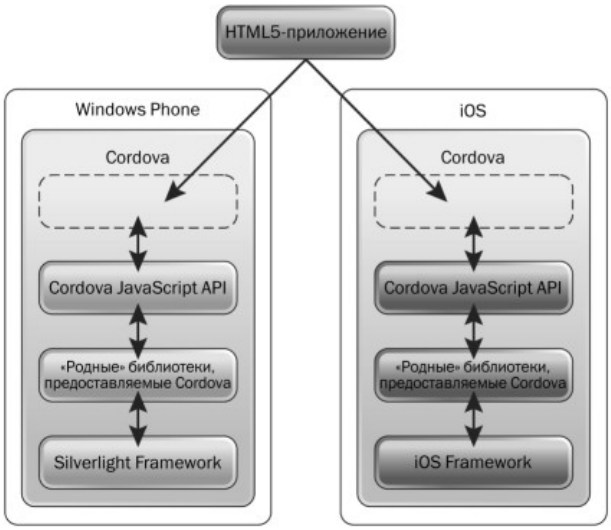

Рис.3. Структурная схема Apache Cordova приложения

Заключение

- Перспектива кроссплатформенных разработок бесспорно положительная: некоторые приложения можно будет писать под несколько платформ сразу без заметного ухудшения их характеристик, что довольно удобно для разработчиков. Кроссплатформенные приложения пишутся на независимом языке, затем компилируются для каждого устройства отдельно.
- Xamarin является средством кроссплатформенной разработки, т.е. приложение, написанное один раз, может быть запущено на различных мобильных платформах, однако код, отвечающий за внешний вид приложения, придется написать для каждой платформы отдельно.
- Главный принцип написанных в Xamarin приложений – они должны выглядеть и вести себя как родные для каждой из мобильных платформ.
- Apache Cordova — технология для создания кроссплатформенных мобильных приложений, использующих HTML5 и JavaScript, а также для разработки приложений Windows Phone.
- Главный принцип написанных в Apache Cordova приложений – использование родного элемента управления ОС - «браузер» для рендеринга контента (HTML5/ JavaScript) вашего приложения.

Список литературы

1. DIY: мобильное приложение для бизнеса. Режим доступа: http://russia.ibuildapp.com/diy (дата обращения 21.01.2015).
2. Кроссплатформенное мобильное приложение. Режим доступа: http://wiki.soloten.com/ (дата обращения 21.01.2015).
3. Шейн Черч. Разработка гибридных веб-приложений, способных использовать аппаратные средства мобильных устройств. Электрон. журн. MSDN Magazine. Режим доступа: https://msdn.microsoft.com/ru-ru/magazine/hh852592.aspx (дата обращения 21.01.2015).
4. Обзор 7 самых популярных кроссплатформенных мобильных фреймворков. Режим доступа: http://habrahabr.ru/post/229559/ (дата обращения 21.01.2015).
5. Кроссплатформенная разработка: анализ вариантов. Режим доступа: http://www.cnews.ru/reviews/new/mobilnye_prilozheniya_dlya_biznesa_2013/articles/krossplatformennaya_razrabotka_analiz_variantov/ (дата обращения 21.01.2015).
6. Гибридные приложения: новое поколение web-приложений. [Электронный ресурс] - Режим доступа: (дата обращения 21.01.2015).
7. Кроссплатформенная разработка. Режим доступа: http://apptractor.ru/develop/cross-platform-development (дата обращения 21.01.2015).
8. Подробно о Xamarin. Режим доступа: http://habrahabr.ru/post/188130/ (дата обращения 21.01.2015).
9. Колин Эберхардт. Разработка HTML5-приложений Windows Phone с применением Apache Cordova. Электрон. журн. MSDN Magazine. Режим доступа: https://msdn.microsoft.com/ru-ru/magazine/hh975345.aspx (дата обращения 21.01.2015).