САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Математико-механический факультет
Кафедра системного программирования

Бумаков Никита Вячеславович

Создание онлайн среды для разработки мобильных приложений

Дипломная работа

Допущена к защите.
Зав. Кафедрой:
д.ф.-м.н., проф. Терехов А. Н.

Научный руководитель:
ст. преп. Брыксин Т. А.

Рецензент:
ст. преп. Ю.В. Литвинов

Санкт-Петербург
2014

SAINT-PETERSBURG STATE UNIVERSITY

Mathematics & Mechanics Faculty

Chair of Software Engineering

Bumakov Nikita

Creation of online environment for mobile applications development

Graduation Thesis

Admitted for defense.

Head of the chair:

Professor A.N. Terekhov

Scientific supervisor:

assistant T.A. Bryksin

Reviewer:

senior lecturer Y.V. Litvinov

Saint-Petersburg

2014

Оглавление

Введение

В последние года особое место на рынке информационных технологий занимают мобильные устройства, число их пользователей постоянно растет. Более того, по статистике аналитической компании Gartner[1] в 2013 году объем продаж смартфонов впервые превысил отметку в 50% среди средств сотовой связи. В связи с этим растет спрос на создание мобильных приложений под различные платформы.

Даже самые несложные приложения требуют определенных навыков программирования, недоступных многим рядовым пользователям мобильных устройств. Поэтому возникла идея создать конструктор приложений, который упростил бы создание мобильных приложений так, чтобы даже относительно сложные приложения можно было создавать без участия программиста.

Со временем начали появляться различные визуальные конструкторы мобильных приложений. Такие конструкторы позволяют создавать мобильные приложения без написания исходного кода самого приложения. Недостатком их является ограниченные возможности по заданию логики. Создание визуального редактора логики - это трудоемкая задача. На данный момент еще не существует средств, дающих возможность наделить приложения сложной бизнес-логикой без непосредственного программирования. Проект QReal:Web направлен на исследование возможности создания такого средства.

На данный момент существует множество платформ, для каждой из которых нужно писать отдельное приложение. В этой связи набирает популярность создание кросс-платформенных приложений на основе технологии HTML/JavaScript. Поэтому QReal:Web направлен на создание не только на создание приложений для конкретной платформы, но и на создание кроссплатформенных HTML приложений.

[1] http://www.gartner.com/newsroom/id/2665715

1. Постановка задачи

Целью данной дипломной работы является создание прототипа онлайн-среды для разработки мобильных приложений, позволяющей визуально задавать внешний вид и логику приложения.

Для достижения этой цели был сформирован следующий набор задач.

- Проанализировать существующие решения, позволяющие создавать мобильные приложения с помощью визуального редактора.
- Разработать общую архитектуру сервиса визуального создания мобильных приложений.
- Разработать архитектуру онлайн дизайнера форм мобильного приложения.
- Разработать прототип веб-сайта, содержащий среду разработки и обеспечивающий основной функционал работы с ней.
- Обеспечить возможность интеграции с генератором приложений под платформу Android и визуальным дизайнером логики, разрабатываемыми в работах Захарова Владимира и Агеева Дениса.
- Обеспечить создание и генерацию кроссплатформенных приложений по описаниям их визуального представления.

2. Предметная область

2.1. Обзор существующих решений

На текущий момент существует несколько онлайн-сервисов для разработки мобильных приложений. Среди них: AppInventor, iBuildApp, Codiqa и т.д. С помощью этих сервисов можно создавать приложения для мобильных телефонов. Эти сервисы реализуют различные подходы к созданию мобильного приложения, такие как создание приложения на основе заранее написанных шаблонов, либо задание интерфейса приложения с простейшей логикой переходов между экранами приложения.

Главным недостатком такого подхода является невозможность задания нетривиальной логики, например, работа с сенсорами телефона (компас, геолокация, акселерометр и т.д.) или с внешними источниками данных. Кроме того, многие из представленных на рынке сервисов ориентированы на создание приложений только для одной конкретной мобильной платформы.

2.1.1. Создание приложения на основе заранее написанных шаблонов

Одним из ярких представителей конструкторов мобильных приложений, которые реализует идею создания мобильных приложений на основе заранее написанных шаблонов, является iBuildApp. Для этого конструктора реализовано множество типовых элементов, таких как виджеты для показа rss, просмотра фотографий, проигрывания аудиопотока, общения в соц. сетях и т.п. Данный подход позволяет создавать многие типовые приложения, но непригоден для создания приложений с уникальным поведением.

Один из шаблонов конструктора iBuildApp представлен на рис. 3.

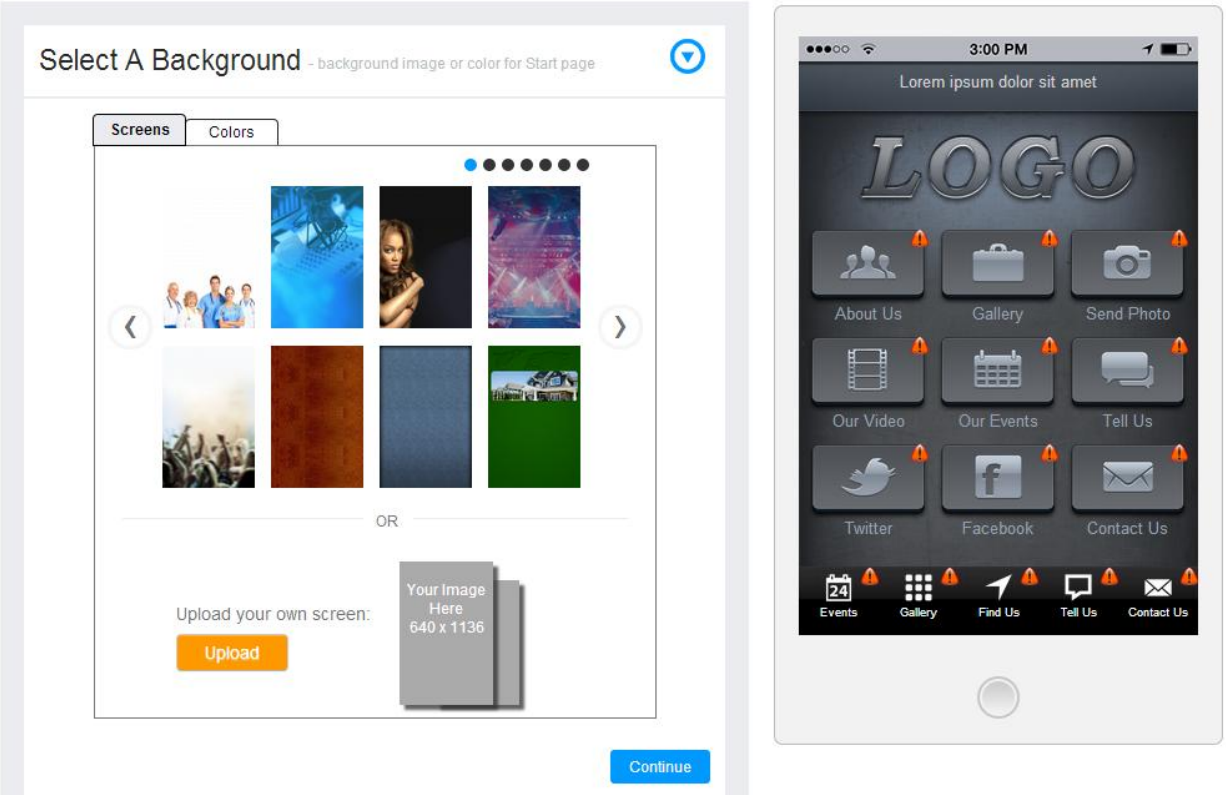

Рис. 1. iBuildApp

2.1.2. Визуальные дизайнеры интерфейса мобильного приложения

Одним из важных классов конструкторов мобильных приложений является визуальный дизайнер интерфейсов. Один из таких конструкторов — Codiqa[2]. Codiqa является сервисом по созданию HTML5 интерфейсов мобильных приложений, на основе библиотеки элементов интерфейса jQueryMobile. Создание интерфейса происходит по принципу перетаскивания элементов интерфейса с палитры на холст приложения и последующем редактировании свойств этих элементов. Логика приложения задается с помощью встроенного редактора кода JavaScript.

Интерфейс дизайнера приложений Codiqa представлен на рис. 2.

[2] https://codiqa.com/

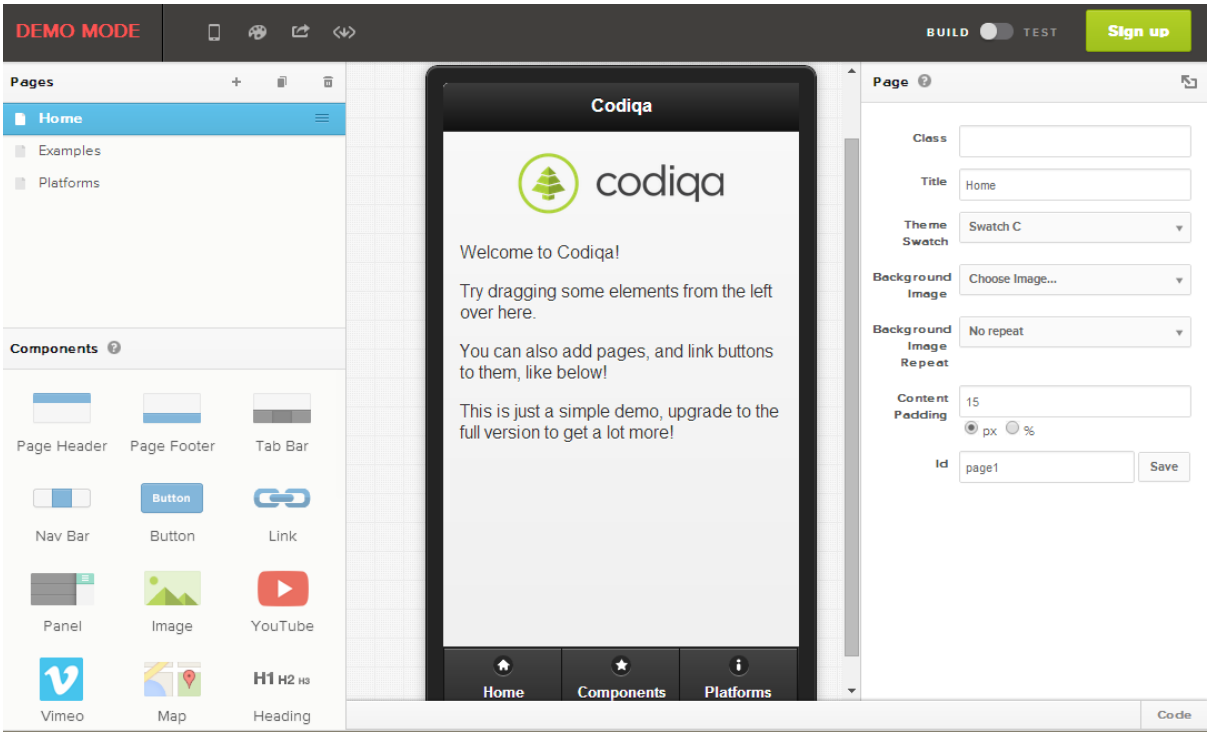

Рис. 2. Codiqa

2.1.3. Конструкторы приложений под конкретные платформы

Существует ряд конструкторов, предназначенных для разработки под одну конкретную платформу. AppInventor[3] – среда разработки мобильных приложений под платформу Android. Среда была разработана в Google Labs, но после закрытия лаборатории была передана в MIT. Отличительной особенностью среды разработки AppInventor является создание приложений под платформу Android, а также наличие Scratch[4]- подобного визуального языка для задания логики мобильного приложения. Такой язык позволяет собирать логику приложения из отдельных блоков, соответствующих конструкциям обычного языка программирования и заранее написанных функций, что по-прежнему остается трудоемким процессом.

Среда разработки состоит из двух частей: визуального редактора и редактора логики. Интерфейс мобильного приложения задается с помощью классического дизайнера с палитрой элементов. Редактором логики является отдельное приложение, скачиваемое на компьютер разработчика.

Интерфейс редактора представлен на рис. 3.

[3] http://appinventor.mit.edu/
[4] http://scratch.mit.edu/

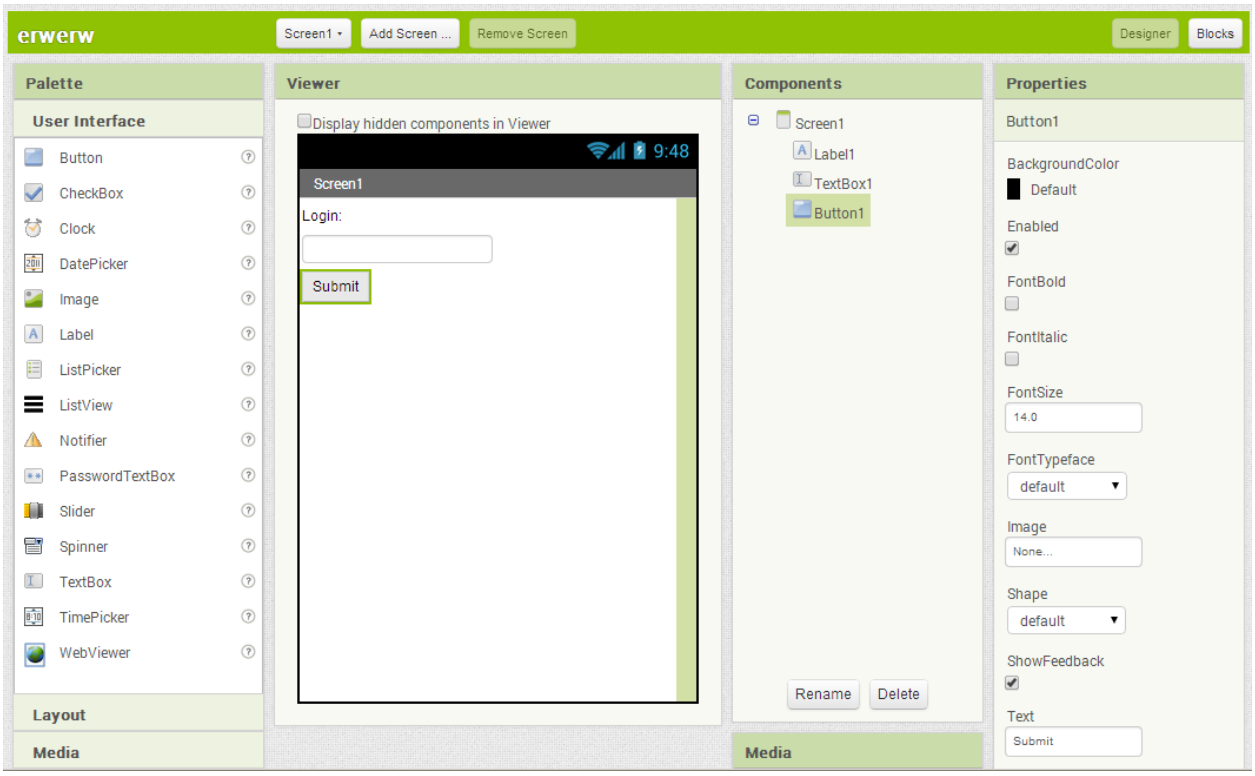

Рис. 3. App Inventor

2.1.4. Выводы

Исходя из анализа текущих конструкторов мобильных приложений можно сделать следующие выводы.

- Несмотря на различные подходы к проектированию приложений у конструкторов, слабым местом остается задание логики мобильного приложения.
- Недостатком некоторых конструкторов является то, что они нацелены на разработку приложений под одну конкретную платформу.
- Ряд конструкторов требуют установки дополнительного программного обеспечения на компьютер пользователя.
- Отсутствие средств для тестирования разработанного мобильного приложения.

Конструктор QReal:Web проектировался таким образом, чтобы избежать этих недостатков.

2.2. Проект QReal:Web

У текущих сервисов был выявлен ряд серьезных недостатков, не позволяющих пользователям создавать полнофункциональные мобильные приложения, не прибегая к написанию кода. Для решения этих проблем на кафедре системного программирования СПбГУ был создан проект QReal:Web. Целью этого проекта является создание среды разработки мобильных приложений, позволяющей быстро и удобно создавать мобильные приложения с нетривиальной логикой.

Среда QReal:Web должна включать в себя средство создания мобильных интерфейсов, визуальный дизайнер логики, онлайн-эмулятор и набор генераторов приложений под различные платформы. В такой системе разработчик мобильного приложения «рисует» пользовательский интерфейс с помощью онлайн-дизайнера, а логику «собирает» из элементов библиотеки готовых заготовок сервисов и привязывает к разработанному пользовательскому интерфейсу.

В 2013 году были написаны прототипы редактора интерфейса и логики, эмулятора и генераторов под платформы Android и Windows Phone 8. Эти работы были первыми исследованиями в области визуального создания мобильных приложений в рамках проекта QReal:Web. В этих работах остался ряд нерешенных проблем, связанных с особенностями разработки веб-приложений и проектированием архитектуры такого сервиса, но тем не менее, создание прототипов дало ценный опыт.

Интерфейс прототипа визуального редактора интерфейса и логики показан на рис. 4. Слева от экрана мобильного приложения находится палитра с визуальными элементами мобильного интерфейса, используемых при создании приложений. Справа располагается scratch-подобный редактор логики приложения.

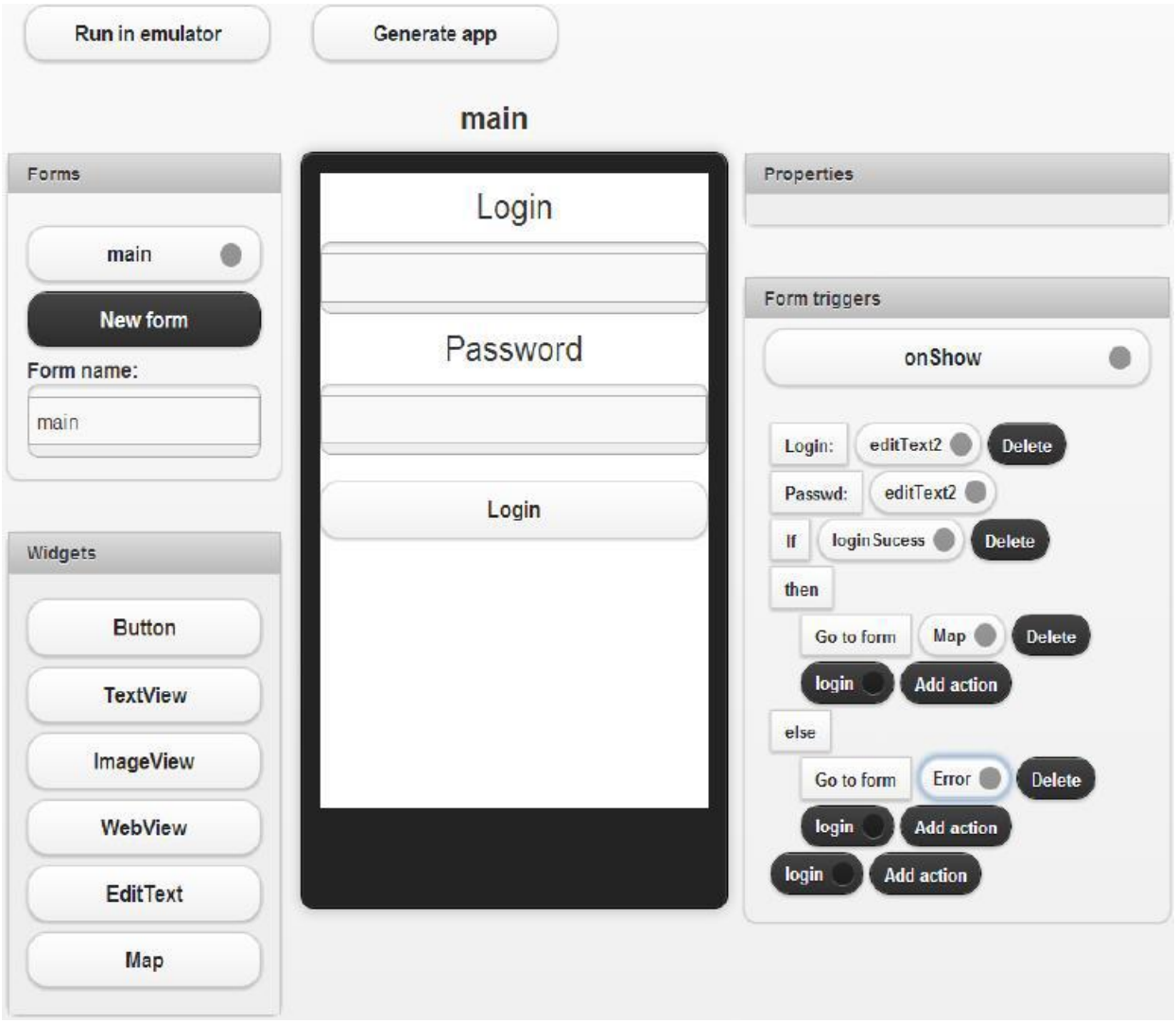

Рис. 4. Прототип дизайнера мобильных интерфейсов

2.3. Обзор используемых инструментов

Проект QReal:Web позиционируется как онлайн-сервис, это накладывает ограничения на используемые технологии. Основным языком разработки браузерных приложений является JavaScript [1], либо языки, транслируемые в него. Разработка на чистом языке JavaScript сопровождается определенными трудностями, такими как: динамическая типизация, которая вызывает множество регрессионных ошибок, прототипная ориентированность языка, отсутствие модульности (нет ни модулей, ни классов). Это делает разработку на этом языке весьма трудоемким процессом. Коме того

результат выполнения программы может отличаться в различных браузерах, это связано с использованием браузерами различных интерпретаторов языка JavaScript.

В связи с описанными выше проблемами был выбран ряд инструментов, позволяющий решать данные задачи: TypeScript[5], jQuery[6], jQueryMobile[7].

2.3.1. TypeScript

Для упрощения разработки на JavaScript различными компаниями предлагались различные решения. В 2012 году компанией Microsoft был представлен новый язык TypeScript [5], позиционируемый как средство разработки веб-приложений и значительно расширяющий возможности JavaScript.

Отличительной особенностью TypeScript'а от других транслируемых в JavaScript языков является полная совместимость с синтаксисом JavaScript. То есть любой корректный код на JavaScript также является корректным кодом и на TypeScript. . Кроме того TypeScript добавляет возможность явного определения типов – статическую типизацию, поддержку полноценных классов как в традиционных объектно-ориентированных языках, а также добавляет поддержку модулей. Это позволяет масштабировать разработку сложных JavaScript приложений.

2.3.2. jQuery

jQuery [2] — это JavaScript библиотека, облегчающая взаимодействие JavaScript и Html. Основные преимущества библиотеки:

- удобный интерфейс для работы с AJAX;
- упрощенная работа с элементами DOM;
- решение проблем совместимости между браузерами.

[5] http://www.typescriptlang.org/
[6] http://jquery.com/
[7] http://jquerymobile.com/

2.3.3. jQueryMobile

Интерфейс мобильных приложений значительно отличается от интерфейса обычных приложений. На текущий момент существует несколько библиотек, позволяющих создавать мобильные интерфейсы с использованием HTML5. Наиболее популярной среди них является библиотека jQueryMobile. jQueryMobile — сенсорно-ориентированный мобильный фреймворк, разрабатывается командой jQuery. jQueryMobile содержит множество элементов мобильных интерфейсов, необходимых при разработке мобильного приложения.

2.3.4. PhoneGap

Помимо задания интерфейса мобильного приложения требуется иметь возможность задания логики. Для HTML5 приложений логика задается с помощью кода на JavaScript. Но также необходимо иметь доступ к базовым сервисам телефона (например геолокация, компас, камера и т.п.). Эти задачи решаются с помощью инструментария с открытым исходным кодом PhoneGap[8]. Данный инструментарий позволяет создавать полнофункциональные мобильные приложения, используя JavaScript, HTML5 и CSS3, без необходимости знания «родных» языков программирования для мобильных платформ.

[8] http://phonegap.com/

3. Архитектура сервиса

На основе анализа преимуществ и недостатков уже существующих сервисов были составлены следующие требования к системе разработки мобильных приложений.

- Визуальное создание интерфейсов мобильного приложения.
- Визуальное задание логики приложения.
- Тестирование создаваемых приложений в браузере.
- Генерация мобильных приложений под конкретные платформы.
- Генерация кроссплатформенных мобильных приложений.
- Работа в различных браузерах.
- Работа в облачной среде (например, в инфраструктуре Windows Azure).

Кроме того среда разработки должна обладать простым интерфейсом, понятным для обычных пользователей без навыков программирования.

Такая система может быть реализована с помощью набора обособленных компонент. Часть из них представляют клиентские программы, выполняемые в браузере пользователя, другая выполняется на сервере. Для обеспечения взаимодействия компонент сервиса используется сериализация состояния редактора в формате JSON. Схема взаимодействия компонент приложения показана на рис. 5.

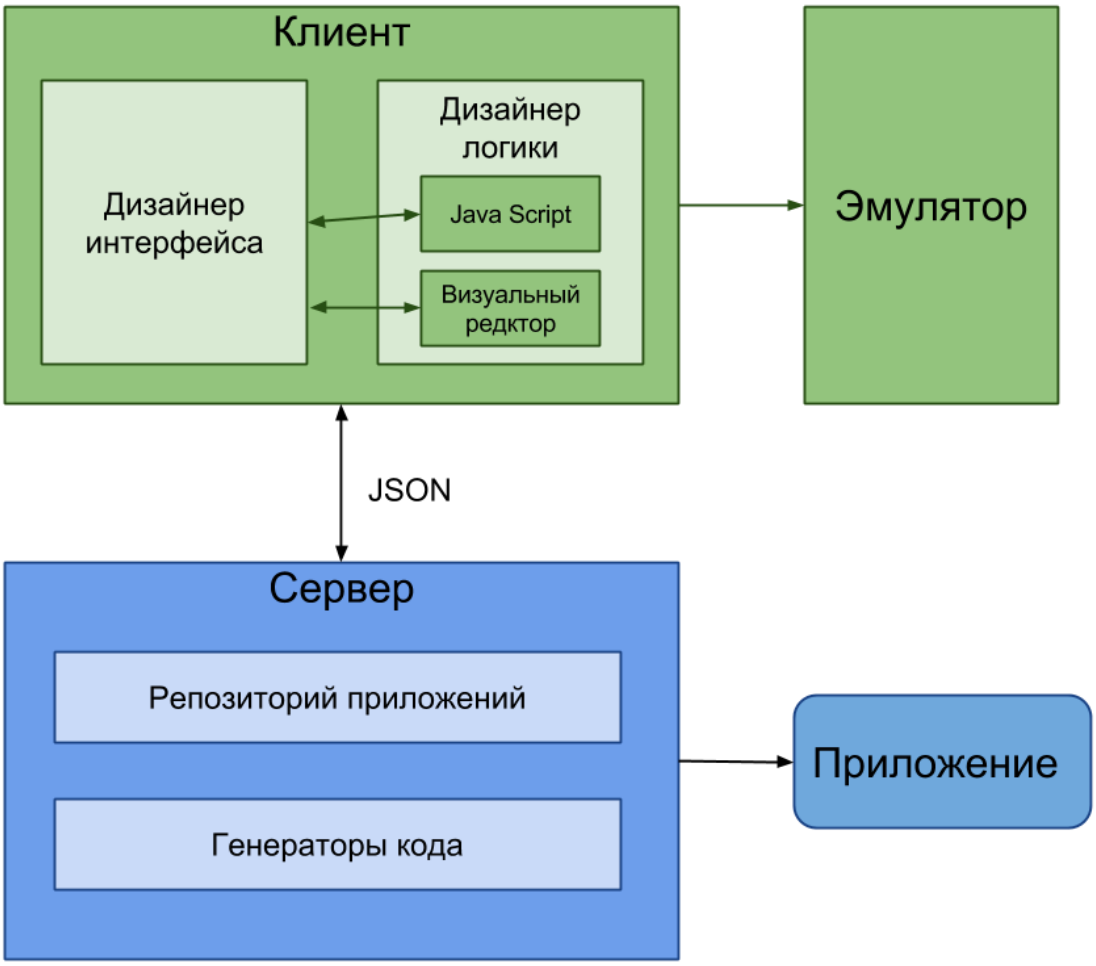

Рис. 5. Общая архитектура системы

Кратко опишем каждую из компонент системы.

- Дизайнер интерфейса.

Данный модуль представляет инструмент для создания интерфейса мобильного приложения. Дизайнер состоит из палитры элементов интерфейса, холста и редактора свойств элементов. Палитра содержит в себе элементы мобильных интерфейсов, таких, как кнопки, меню, поля ввода, текстовые поля, картинки, и т.д. На холст можно вытаскивать мышью элементы из палитры или перемещать уже существующие элементы. Редактор свойств элементов позволяет менять внешний

вид составляющих частей интерфейса, например, изменять тему страницы мобильного приложения.

- Дизайнер логики.

  С помощью данного дизайнера пользователь может задать нетривиальную логику приложения с использованием наглядных и понятных непрофессионалу визуальных языков. Такой подход позволяет наглядно разрабатывать приложения без кодирования на текстовых языках, что позволяет делать это даже людям, не владеющим программированием. Палитрой такого дизайнера является набор элементов интерфейса, созданных в дизайнере интерфейса, и набора сервисов, таких как доступ к сенсорам телефона или работа с внешними сервисами. Прототип такого дизайнера разработан в рамках курсовой работы студента 345 группы Агеева Дениса.

  В случае отсутствия возможности задать логику с помощью визуальных языков пользователь имеет возможность задать логику HTML5 приложения с помощью кода на JavaScript. Для этой цели в среде предусмотрен редактор кода JavaScript.

- Отладочный эмулятор.

  Эмулятор позволяет запустить создаваемое в редакторе приложение без компиляции приложения. Эмулятор создает тестовое окружение для HTML5 приложения, предоставляя интерфейс задания тестовых данных для сенсоров телефона и обеспечивая работу с внешними сервисами. Применяется для упрощения отладки интерфейса и логики приложения.

- Сервер

  Сервер обеспечивает взаимодействие между компонентами системы, также на сервере хранятся созданные пользователем приложения.

- Генераторы приложений

  С помощью генератора приложений создаются готовые мобильные приложения, которые могут быть установлены на телефон или размещены в магазине приложений. В зависимости от используемого генератора может быть создано как приложение под конкретную платформу, так и кроссплатформенное.

Кроссплатформенные приложения создаются на основе библиотеки PhoneGap. Прототип генератора приложений под платформу Android разработан в рамках курсовой работы студента 345 группы Владимира Захарова.

4. Редактор интерфейса

Одной из основных компонент среды разработки мобильных приложений является визуальный редактор интерфейса. В ходе данной работы разработан редактор, позволяющий задавать интерфейс мобильного приложения. Внешний вид редактора представлен на рис. 6.

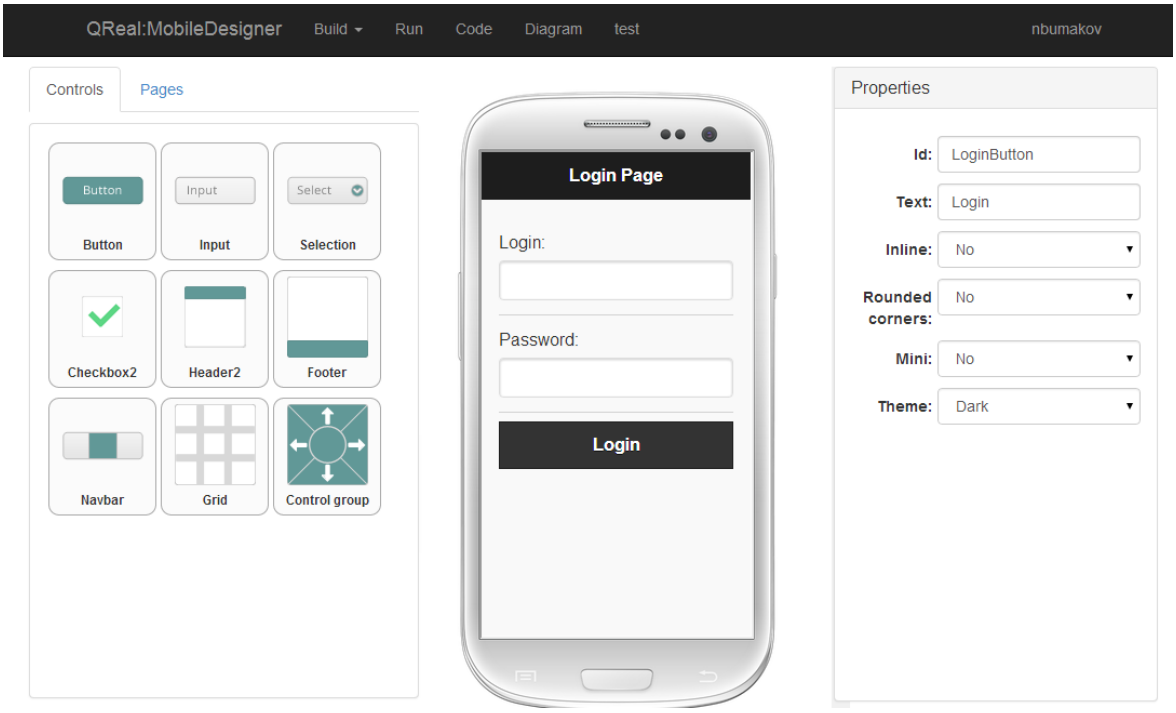

Рис. 6. Интерфейс редактора интерфейса.

4.1. Функционал редактора интерфейсов

Основной рабочей областью является холст с экраном мобильного приложения. На холст перетягиваются элементы из палитры элементов, расположенной слева. Созданный элемент можно отредактировать путем изменения его свойств в редакторе в правой части экрана или изменить его положение путем перетаскивания по холсту.

Так как приложения обычно состоят более, чем из одного экрана, то предусмотрена возможность создания нескольких страниц. Управление создаваемыми страницами происходит во вкладке “Pages”. Если приложение содержит более одной страницы, то стартовой будет первая из них.

Мобильные приложения могут состоять из большого количества различных компонент. При создании дизайнера был выбран и реализован следующий набор основных компонент мобильного интерфейса, который может быть расширен в дальнейшем:

- Button. Кнопка — основной управляющий элемент, к которому привязывается логика обработки нажатия.
- TextView. Текстовое поле, для вывода текстовой информации.
- Input. Поле текстового ввода.
- ImageView. Отображает картинку по заданному URL.
- WebView. Встроенный в приложение браузер позволяет показать веб страницу по заданному URL.
- Map. Позволяет использовать в приложении карты, предоставляемые сервисом Google Map.

Данный набор элементов позволяет создавать простые интерфейсы приложения и обеспечивает необходимый набор возможностей, для тестирования генераторов приложения и визуального редактора логики.

4.2. Реализация редактора интерфейсов

Редактор интерфейса представляет собой веб-приложение. Для написания использовался язык TypeScript, который позволил построить классическую ООП архитектуру приложения.

4.2.1. Взаимодействие между отдельными компонентами дизайнера

При реализации возникла необходимость использования различных библиотек для задания интерфейса редактора и мобильного приложения, поскольку иначе схожие элементы интерфейса из разных библиотек начинают конфликтовать. Для решения такой проблемы было решено разбить дизайнер на две отдельные компоненты: дизайнер и холст мобильного приложения. Холст мобильного приложения вынесен в отдельный компонент, который помещается в iframe[9]. Такой подход дает возможность использовать в раздельных

[9] <iframe> - это тег, который находится внутри обычного html документа, позволяя загружать любые другие независимые html документы

компонентах различные библиотеки. Для создания интерфейса дизайнера использовалась библиотека Bootstrap[10] [6], а для интерфейса мобильного приложения используется библиотека jQueryMobile.

Код этих компонент работает раздельно друг от друга, так как контексты выполнения для них различны. Для обеспечения взаимодействия этих компонент был расширен родительский класс window[11]. В него был добавлен класс Application, содержащий контроллеры обоих компонент. Схема представлена на рис 7.

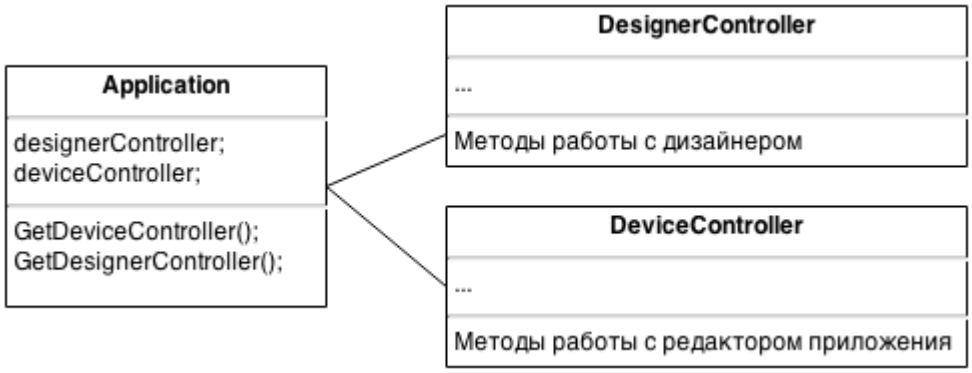

Рис. 7. Схема взаимодействия компонент системы.

При инициализации каждая компонента сохраняет в этом классе ссылки на свои контроллеры. При необходимости обратиться одному компоненту к другому нужно просто вызвать соответствующий метод контроллера нужного компонента.

4.2.2. Иерархия элементов мобильного интерфейса

При создании редактора интерфейса возникла задача создания двух типов элементов графического интерфейса: для конечного мобильного приложения и для самого дизайнера интерфейса. Данные типы компонент различаются лишь тем, что компоненты для дизайнера обладают рядом дополнительных свойств, таких как редактирование свойств, перетаскивание по холсту и т.п.

Благодаря языку TypeScript, который вносит элементы ООП в классический JavaScript, есть возможность построить ООП архитектуру, в которой элементы для дизайнера наследуются от элементов мобильного приложения.

[10] http://getbootstrap.com/
[11] Объект window представляет открытое окно браузера

Для создания элементов применяются две фабрики. Для обоих фабрик выделен общий интерфейс, которой эти фабрики наследуют, причем фабрика элементов для дизайнера расширяет фабрику мобильных элементов интерфейса, добавляя специфическое для дизайнера поведение.

Схема построения такой архитектуры показана рис. 8.

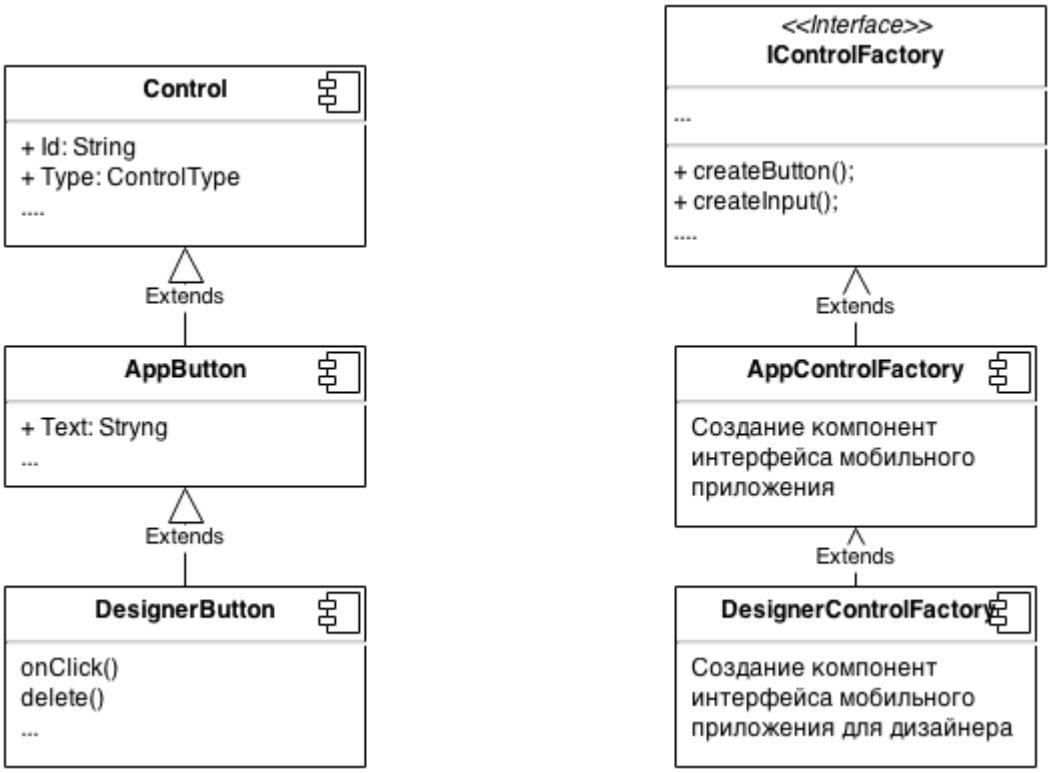

Рис. 8. Иерархия мобильных элементов

5. Функционал прототипа среды разработки

Помимо дизайнера мобильных интерфейсов реализован прототип среды разработки. Среда представляет собой веб-сайт, который содержит все компоненты системы и обеспечивает взаимодействие между ними. Сайт реализован на C# с использованием технологии ASP.NET MVC 5. С деталями использования данного языка и технологии можно ознакомиться в книгах Герберта Шилдта [3] и Стивена Сандерсона [4].

.

5.1. Создание проекта пользователя

Для работы с сервисом генерации мобильных приложений пользователь должен зарегистрироваться в системе. После регистрации пользователю становится доступна возможность создания нового проекта. Данные о зарегистрированных пользователях и об их проектах хранятся в репозитории на основе базы данных Microsoft SQL Server 2012. Сами проекты создаются и хранятся в файловой системе сервера.

При создании нового проекта на сервере создается инфраструктура, соответствующая выбранному типу приложения. На текущий момент поддерживается возможность создания кроссплатформенных HTML5 приложений, основанных на библиотеке PhoneGap и приложений под платформу Android.

5.2. Задание логики приложения

Помимо дизайнера интерфейсов среда разработки содержит визуальный редактор логики и обычный редактор кода на JavaScript. Редактор кода может потребоваться в случае, если визуальный редактор не поддерживает задание необходимой логики, либо если для создания приложения пользователю удобнее использовать обычный редактор кода.

При вызове редактора логики в редактор передается информация об имеющихся в создаваемом приложении элементах интерфейса. Эта информация передается с помощью JSON объекта, содержащего список элементов с их id и типом.

Визуальный язык дизайнера интерфейса состоит из сервисов, поддерживаемых телефоном, либо внешних, доступных с телефона, и элементов интерфейса, взаимодействующих с этими сервисами.

Прототип визуального дизайнера логики представлен на рис. 9.

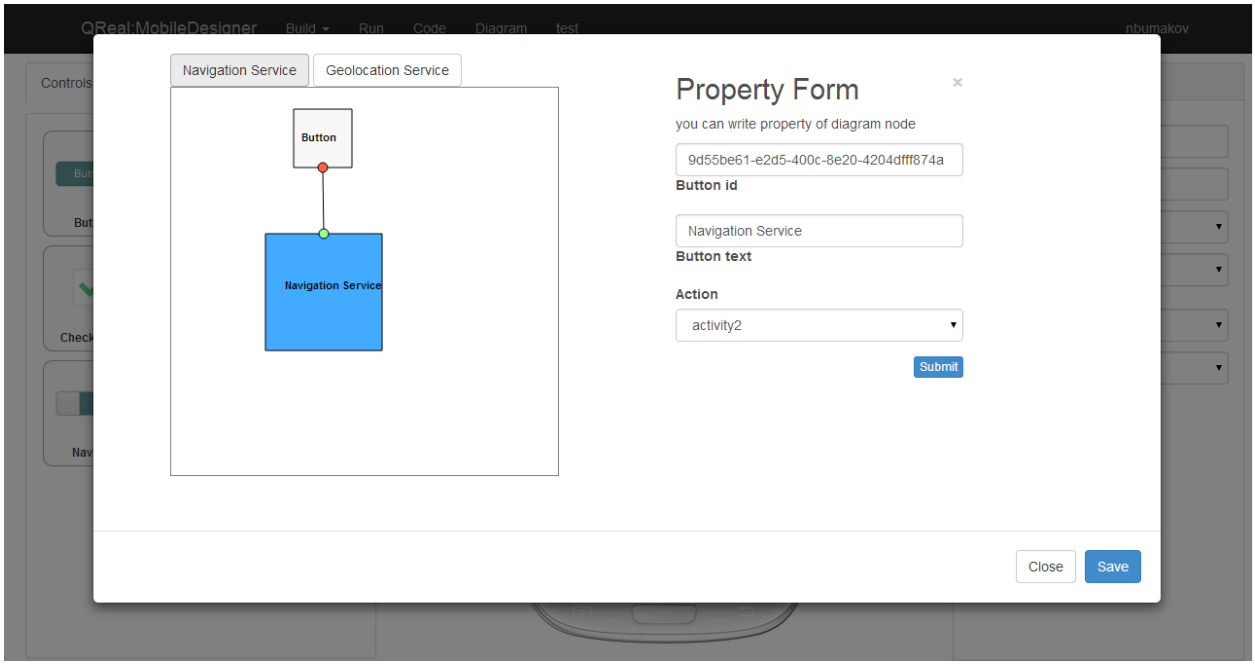

Рис. 9. Интерфейс прототипа визуального редактора

5.3. Отладка на эмуляторе

Разработка мобильного приложения требует проверки промежуточного результата на эмуляторе. Для этих целей был создан прототип эмулятор приложения. По кнопке “Run” главного окна среды разработки на сервер отправляется запрос, содержащий HTML с интерфейсом приложения, а также код JavaScript и CSS. На сервере эти данные собираются в единое приложение, которое показывается пользователю.

Такой подход позволяет проверить работу приложения. связанную с внутренней логикой или работу с внешними сервисами, но не позволяет проверить взаимодействие с сенсорами телефона. Для эмуляции работы с сенсорами телефона требуется реализовать библиотеку, которая будет обладать таким же интерфейсом, как и в библиотеке Apache Cordova[12] (часть фреймворка PhoneGap), возвращая тестовые данные на запросы к сенсорам телефона. А также интерфейс для задания этих тестовых данных (например координаты для геолокационного сервиса).

Реализация такой библиотеки возможна в отдельной работе, в рамках проекта QRealWeb.

[12] http://cordova.apache.org/

6. Интеграция с другими компонентами

Интеграция с другими компонентами системы осуществляется с помощью сериализации внутреннего представления. Сериализатор обходит созданную модель приложения, создавая при этом JSON с описанием интерфейса приложения. В сериализованном виде модель интерфейса создаваемого приложения может быть передана в визуальный редактор логики, генератор приложения или просто сохранена на сервере.

Из окна среды разработки возможно вызвать визуальный редактор логики, при этом действии внутреннее представление сериализуется в JSON и отправляется визуальному редактору логики. Результат работы визуального дизайнера также представляет JSON с описанием заданной логики. При генерации мобильного приложения под конечную платформу на сервер отправляются JSON файлы с описанием логики и внешнего вида приложения, по которым генерируется файл с приложением для нужной платформы.

7. Генерация кроссплатформенных приложений

Кроссплатформенные мобильные приложения разрабатываются с помощью технологий HTML и JavaScript. Графический интерфейс приложения задается при помощи HTML, логика задается кодом на JavaScript. Технология PhoneGap позволяет JavaScript коду взаимодействовать с сенсорами телефона.

Создание визуальных элементов на холсте дизайнера интерфейса происходит в два этапа. На первом этапе создаются элементы интерфейса для мобильного приложения, на втором эти компоненты расширятся логикой, специфической для дизайнера интерфейса, например, вызов редактора свойств элемента. Такой подход позволяет параллельно с редактированием интерфейса мобильного приложения в дизайнере создавать компоненты интерфейса конечного мобильного приложения.

Таким образом, приложение на сервере генерируется на основе двух файлов: один в формате HTML, другой — в JSON. Первый файл создается при проектировании интерфейса приложения, а второй получается от визуального дизайнера логики.

Апробация

В ходе работы была проведена апробация, целью апробации является проверить возможность использования дизайнера интерфейса и визуального редактора логики при создании мобильного приложения. Апробация проведена на примере создания простого приложения и приложения с логикой взаимодействия с сенсорами телефона.

Простейший пример приложения

Примером простейшего приложения является приложение “Визитка”. Это приложение, отображающее общую информацию об организации, контакты и местоположение предприятия на карте. В таком приложении используются заранее заданные в редакторе интерфейсов данные и переходы между страницами, реализованные с помощью сервиса переходов.

На рис. 10 показано разрабатываемое приложение в интерфейсе дизайнера, на рис. 11 показаны скриншоты получаемого приложения.

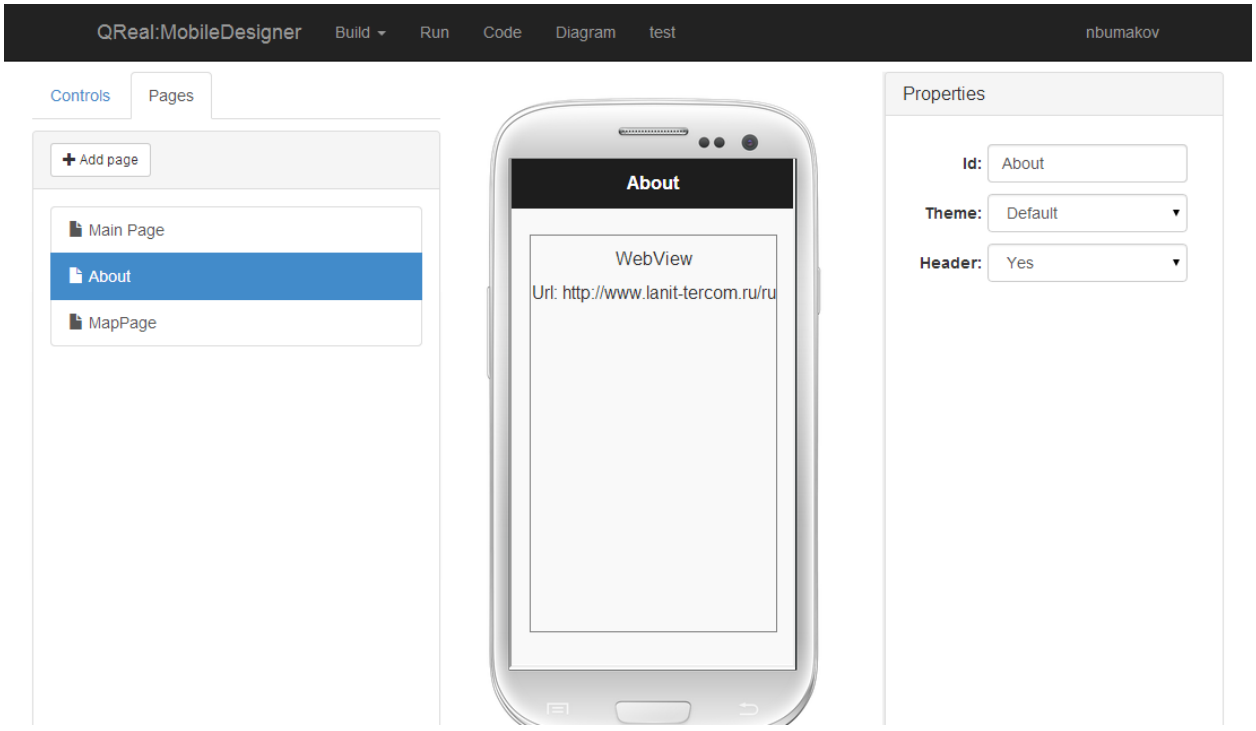

Рис. 10. Интерфейс среды разработки

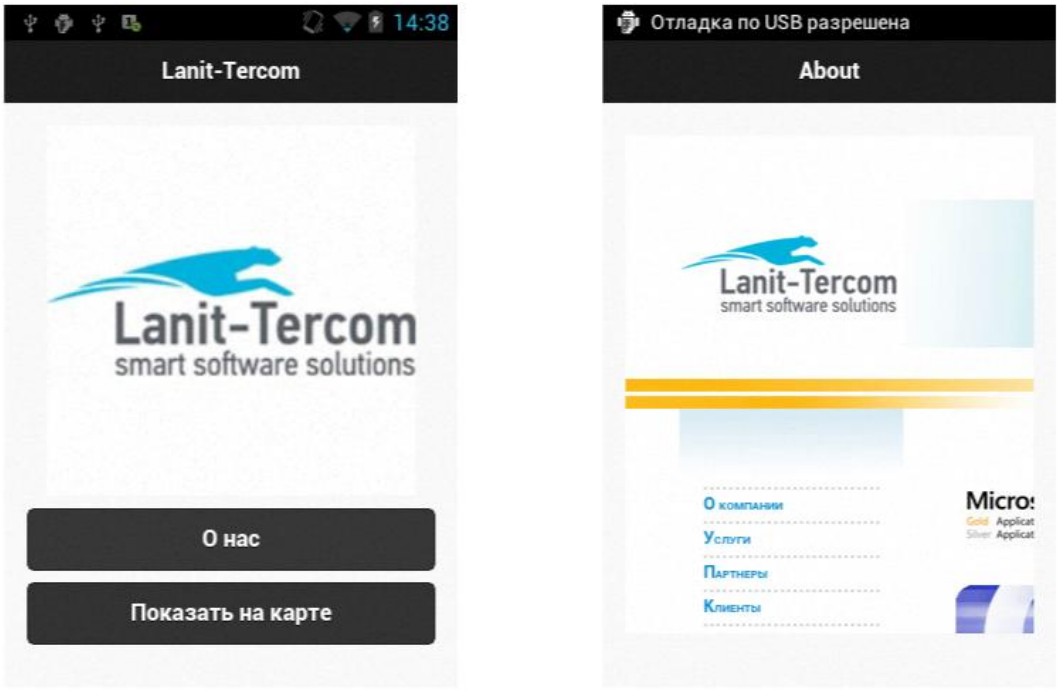

Рис. 11. Интерфейс кроссплатформенного приложения-визитки

Пример взаимодействия с сенсорами телефона

Для проверки взаимодействия с сенсорами телефона было создано приложение, показывающее местоположение телефона на карте. Запрос к GPS сервису телефона происходит по нажатию на кнопку. Логика такого приложения может быть задана с помощью прототипа визуального редактора при помощи схемы на рис 10.

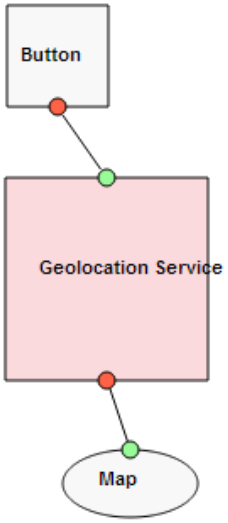

Рис. 10. Схема обновления положения пользователя

На рис. 11. изображен пример работы такой программы.

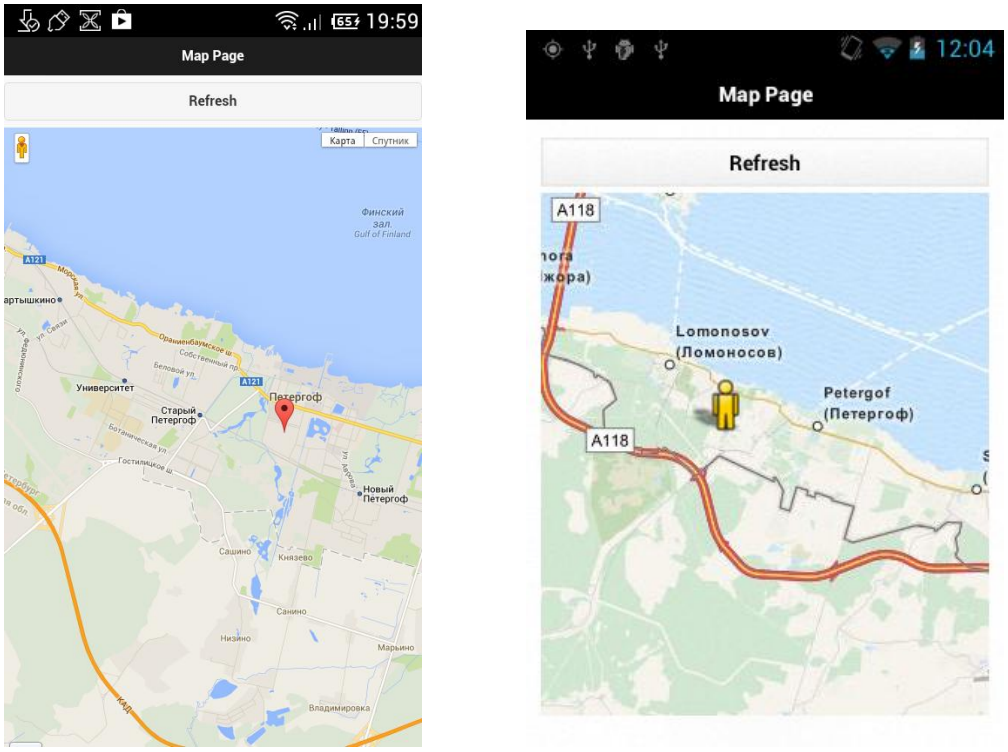

Рис. 11. Пример работы мобильного кроссплатформенного приложения (слева) и приложения на платформе Android (справа), использующих геолокацию

Результаты

В ходе работы были проанализированы существующие на данный момент подходы к визуальному созданию мобильных приложений. Был разработан визуальный дизайнер интерфейсов, позволяющий создавать интерфейсы кроссплатформенных мобильных приложений. Также был разработан прототип среды разработки, на основе которого возможно создание мобильных приложений с применением средств визуальной разработки.

Ниже перечислены результаты данной дипломной работы.

- Изучены существующие подходы к визуальному созданию мобильных приложений, в ходе которого были выявлены недостатки текущих решений.
- Спроектирована архитектура среды разработки.
- Создан онлайн дизайнер, поддерживающий создание кроссплатформенных JQuery Mobile приложений.
- Сделан прототип сайта, поддерживающий регистрацию, создание и сборку проекта.
- Проведена интеграция с генератором приложений под платформу Android и визуальным редактором логики.
- Реализована возможность сборки кроссплатформенных приложений с использованием технологии PhoneGap.

Дальнейшее развитие

Созданный в ходе данной работы дизайнер мобильных интерфейсов и прототип среды разработки мобильных приложений может использоваться для дальнейших исследованиях в рамках проекта QReal:Web.

- Исследование различных предметно-ориентированных языков (DSL, Domain Specific Languages), которые могут быть использованы для создания логики мобильных приложений.
- Исследование возможностей создания генераторов приложений под конкретные платформы.

Кроме того функционал текущей среды разработки может быть расширен следующими возможностями:

- Полнофункциональный эмулятор приложений в браузере, поддерживающий эмуляцию основных сервисов телефона (геолокация, файловая система, звонки и т.д.)
- Контроль версий разрабатываемого приложения.
- Совместная разработка одного приложения несколькими пользователями.

Список литературы

[1] Дэвид Флэнаган, JavaScript. Подробное руководство // Издательство “Символ-Плюс”, 2012, 1080С

[2] Бер Бибо, Иегуда Кац, jQuery. Подробное руководство по продвинутому JavaScript // Издательство “Символ-Плюс”, 2011, 623С

[3] Герберт Шилдт, C# 4.0. Полное руководство // Издательство “Вильямс”, 2011, 1056С

[4] Стивен Сандерсон, ASP.NET MVC Framework с примерами на C# для профессионалов // Издательство “Вильямс”, 2010, 557С

[5] Maharry Dan, TypeScript Revealed // Издательство “Apress”, 2013, 104С

[6] Jake Spurlock, Bootstrap. Responsive Web Development // Издательство “O'Reilly Media”, 2013, 128С

[7] JQuery, URL: http://jquery.com/

[8] TypeScript, URL: http://www.typescriptlang.org/

[9] Bootstrap, URL: http://getbootstrap.com/

[10] JQueryMobile, URL: http://jquerymobile.com/

[11] PhoneGap, URL: http://phonegap.com/