Содержание

Предисловие

Свое первое одностраничное веб-приложение (single page web application – SPA) я написал в 2006 году, хотя тогда оно еще так не называлось. Для меня это стало заметным событием. До того я занимался низкоуровневым программированием ядра Linux, а также параллельными и распределенными вычислениями, а пользовательский интерфейс сводился к простой командной строке. Получив в 2006 году постоянную преподавательскую работу в Университете Сан-Франциско, я затеял амбициозный проект в области распределенных вычислений, получивший название River (http://river.cs.usfca.edu). Там был нужен интерактивный графический интерфейс для управления распределенными машинами и отладки системы.

Алекс Рассел (Alex Russell) как раз тогда придумал термин «комета» (comet)[1], и мы с воодушевлением решили использовать эту технологию для реализации интерфейса в веб-браузере. Возникли сложности при попытке организовать взаимодействие в реальном времени с помощью JavaScript. Худо-бедно все работало, но не так эффективно, как мы надеялись. Проблема была в том, что нам почти все приходилось разрабатывать самостоятельно, так как имеющихся сегодня библиотек и технических приемов тогда еще не было и в помине. Например, первая версия jQuery вышла как раз в тот год, но позже.

В июле 2011 года я был директором по разработкам в компании SnapLogic, Inc. (http://snaplogic.com), а Майк Миковски был принят на должность архитектора пользовательского интерфейса. Мы работали в команде, которая проектировала продукт для интеграции данных следующего поколения. Вместе с Майком мы провели бессчетные часы за обсуждением принципиальных вопросов программной инженерии и проектирования языков. Мы многому научились друг у друга. Майк также показал мне черновики той книги, которую вы сейчас читаете, и именно тогда я узнал о придуманном им и Джошем методе построения SPA. Было понятно, что они разработали несколько поколений коммерческих SPA и воспользовались этим опытом, чтобы отточить технику и архитектурные принципы, сделав их полными, ясными и сравнительно простыми.

---

[1] Comet – любая модель работы веб-приложения, при которой постоянное HTTP-соединение позволяет веб-серверу отправлять данные браузеру без дополнительного запроса со стороны браузера. См. http://ru.wikipedia.org/wiki/Comet_(программирование). – *Прим. перев.*

Со времен проекта River 2006 года ингредиенты одностраничных приложений, работающих в браузере, достигли высокой степени зрелости и теперь, вообще говоря, превосходят сторонние подключаемые модули типа Java или Flash. Существует немало прекрасных книг об этих ингредиентах по отдельности: HTML, CSS, JavaScript, jQuery, NodeJS и HTTP. Но, к сожалению, почти нет книг, описывающих, как объединить их в единое целое.

Эта книга является исключением из правила. В ней подробно излагаются тщательно проверенные рецепты построения неотразимых SPA с помощью JavaScript, применяемого на всех уровнях стека. Авторы делятся опытом, приобретенным и отточенным в ходе разработки многих поколений SPA. Можно сказать, что Майк и Джош совершили немало ошибок, которые вам повторять вовсе не обязательно. Прочитав эту книгу, вы сможете сосредоточиться на цели приложения, а не на его реализации.

Решения, описанные в этой книге, основаны на современных вебстандартах, поэтому они останутся актуальны еще долго и должны работать в самых разных браузерах и устройствах. Как бы мне хотелось, чтобы современные технологии и эта книга существовали в далеком 2006 году, когда мы работали над проектом River. Уж мы бы нашли им применение!

Грегори Д. Бенсон,
профессор факультета информатики
Университета Сан-Франциско

Вступление

С Джошем мы встретились летом 2011 года, когда я искал работу, и он предложил мне место архитектора. Хотя в конечном итоге я решил принять другое предложение, мы подружились и с интересом обсуждали тему одностраничных веб-приложений и будущего Интернета. Как-то раз Джош по наивности предложил вместе написать книгу. Я сдуру согласился, и это решило нашу судьбу на последующие сотни недель. Мы думали, что книга получится тоненькой, не больше 300 страниц. Идея была в том, чтобы постоять за спиной опытного разработчика, создающего SPA промышленного качества с помощью одного лишь JavaScript. Мы собирались использовать только лучшие в своем классе инструменты и приемы, чтобы предложить пользовательский интерфейс мирового уровня. Все наши идеи должны были быть применимы к разработке любого SPA на JavaScript – так, как делается в этой книге, или с помощью иных имеющихся библиотек и каркасов.

После того как черновик книги был опубликован на сайте издательства Manning в рамках программы раннего ознакомления, в первый же месяц поступило более тысячи заказов на книгу. Мы прислушивались к мнениям читателей и беседовали с тысячами разработчиков и авторитетов на разных встречах, в университетах, на отраслевых конференциях – чтобы разобраться, в чем заключается очарование SPA. Из услышанного мы сделали вывод, что существует неутоленная жажда знаний по этому вопросу. Мы поняли, что разработчики мечтают узнать о более совершенных способах создания веб-приложений. Поэтому мы включили дополнительные темы. Например, мы добавили приложение Б, по объему сопоставимое с главой, в котором детально описываем, как настроить автоматизированное тестирование SPA, поскольку многим читателям показалось, что тестирование рассматривается в рукописи недостаточно подробно.

В итоге мы сохранили подход к изложению, основанный на идее подглядывания за опытным разработчиком, и добавили ряд тем, о которых просили читатели. В результате «маленькая книжка» разрослась и почти вдвое превысила первоначально планировавшийся объем. Надеемся, что вам она понравится.

Майкл С. Миковски

Благодарности

Авторы выражают благодарность следующим лицам.

- Джою Бруксу (Joey Brooks), специалисту по подбору персонала, который познакомил нас. Это ты во всем виноват, Джой.
- Джону Резигу (John Resig) и всем разработчикам jQuery за создание фантастической библиотеки – лаконичной, расширяемой и функционально богатой. Благодаря jQuery SPA оказываются быстрее и надежнее, а их разработка доставляет удовольствие.
- Яну Смиту (Ian Smith) за создание и сопровождение TaffyDB, мощного инструмента для манипуляции данными в браузере.
- Нильсу Джонсону (Niels Johnson), известному также как «Spikels», за предложение вычитать наш материал в обмен на раннее ознакомление. Думаю, что от этой сделки выиграли мы, потому что его рецензии оказались поразительно подробными и весьма полезными при окончательном редактировании.
- Майклу Стивенсу (Michael Stephens) из издательства Manning, который помог нам собрать первый черновой вариант и определиться со структурой книги.
- Берту Бейтсу (Bert Bates), который лучше большинства живущих на этой планете знает, как писать технические книги. Он научил нас всегда помнить о том, для какой аудитории мы пишем.
- Карен Миллер (Karen Miller), нашему редактору-консультанту, которая работала вместе с нами почти все время, понукала и нас, и всех остальных участников процесса, не позволяя останавливаться на полпути.
- Бенджамину Бергу (Benjamin Berg), нашему редактору; Джанет Вайль (Janet Vail), редактору по производству, фантастически коммуникабельной женщине, знающей все о том, как подготовить издание к печати; и всем сотрудникам издательства Manning, помогавшим нам в работе над книгой.
- Эрнесту Фридман-Хиллу (Ernest Friedman-Hill), консультанту по техническим иллюстрациям, который подал нам идеи, положенные в основу самых удачных рисунков в этой книге.
- Джону Дж. Райану (John J. Ryan) за скрупулезное техническое редактирование окончательного варианта рукописи непосредственно перед сдачей в печать.
- Всем рецензентам, детально проанализировавшим наш текст и код и предложившим упрощения и улучшения: Анне Эпштейн (Anne Epstein), Чарльзу Энгельке (Charles Engelke), Кэрти-

су Миллеру (Curtis Miller), Даниэлю Бретуа (Daniel Bretoi), Джеймсу Хэтуэю (James Hatheway), Джейсону Качору (Jason Kaczor), Кену Муру (Ken Moore), Кену Римплу (Ken Rimple), Кэвину Мартину (Kevin Martin), Лео Половецу (Leo Polovets), Марку Райалу (Mark Ryall), Марку Торрэнсу (Mark Torrance), Майку Гринхалгу (Mike Greenhalgh), Стэну Байсу (Stan Bice) и Вайятту Барнетту (Wyatt Barnett).

- Тысячам людей, купивших ранний вариант книги, участникам конференций и коллегам, которые побуждали нас оптимизировать представленные в книге решения.

Майк также благодарит:

- Джоша Пауэлла, который предложил мне написать эту книгу. Это были замечательная идея и великолепный опыт, многому научивший меня. Только вот нельзя ли вернуть потраченные годы жизни?
- Грега Бенсона за написание предисловия и за напоминание о том, что это слово пишется «foreword», а не «forward».
- Гаурава Дхиллона (Gaurav Dhillon), Джона Шустера (John Schuster), Стива Гудвина (Steve Goodwin), Джойса Лэма (Joyce Lam), Тима Лайкариша (Tim Likarish) и других сотрудников компании SnapLogic, которые понимают ценность экономности и элегантности решения.
- Анис Икбал (Anees Iqbal), Майкла Лортона (Michael Lorton), Дэвида Гуда (David Good) и других сотрудников компании GameCrush. Разработка продукта в GameCrush поставлена не идеально, но ничего более близкого к идеалу я не встречал.
- Моим родителям за то, что они купили мне компьютер, но отказались покупать к нему какой-нибудь софт. Это стало отличным поводом научиться программировать.
- Всем, кого я забыл упомянуть. Согласно закону Мэрфи, подраздел 8, я наверняка забыл кого-то очень важного, а вспомню только после выхода книги из печати. За это я искренне прошу прощения и надеюсь, что оно будет даровано.

Джош также благодарит:

- Майка Миковски за согласие совместно написать книгу. Я так рад, что мне не пришлось писать всю книгу самому. Тунеядец! Я имел в виду – спасибо большое.
- Луку Пауэлла, моего брата, который имеет мужество не отказываться от своей мечты, создать бизнес и быть самим собой. Он меня вдохновляет.

- Других членов моей семьи и друзей, без которых я не стал бы тем, кто я есть.
- Джона Келли (John Kelly) за то, что он позволил мне закончить книгу, понимая, что для такого дела требуется время. Да еще какое!
- Марку Торренсу (Mark Torrance) за то, что он наставлял меня, когда я создавал квалифицированную команду инженеров, и за свободу, предоставленную, когда я начал писать книгу.
- Уилсону Янгу (Wilson Yeung) и Дэйву Киферу (Dave Keefer), которые подталкивали меня к более глубокому изучению веб-технологий. Вы оказали огромное влияние на мою карьеру и знания в области программной инженерии.

Об этой книге

Задумывая эту книгу, мы собирались посвятить примерно две трети разработке клиентской части SPA. А в оставшейся трети мы планировали рассказать о веб-сервере и службах, необходимых для работы SPA. Но никак не могли выбрать конкретный веб-сервер. Нам доводилось писать десятки серверных веб-приложений – традиционных и одностраничных – с применением Ruby/Rails, Java/Tomcat, mod_perl и других платформ, но у всех были какие-то недостатки, особенно в плане поддержки SPA, заставлявшие нас желать большего.

Недавно мы перешли на «чистую» JavaScript-платформу: Node.js в качестве веб-сервера и MongoDB в качестве базы данных. Хотя кое-какие трудности имеются, мы считаем такое сочетание весьма привлекательным и раскрепощающим. Преимущества единого языка и общего формата данных обычно намного перевешивают потерю некоторых языковозависимых возможностей, присутствующих в «полиглотическом» стеке.

Мы решили, что знакомство с «чистым» JavaScript'овым стеком принесет нашим читателям набольшую пользу, поскольку нам неизвестна никакая другая книга, в которой показано, как все эти компоненты собрать воедино. И мы полагаем, что эта платформа будет набирать популярность и станет одной из самых распространенных платформ для создания одностраничных приложений.

Структура книги

Глава 1 содержит введение в одностраничные приложения. Определяется, что такое JavaScript SPA, и проводится сравнение с другими видами SPA. SPA сравниваются с традиционными веб-сайтами, обсуждаются достоинства, возможности и проблемы, возникающие при использовании SPA. По ходу главы описывается постепенная разработка работоспособного SPA.

В главе 2 рассматриваются средства JavaScript, необходимые для создания SPA. Поскольку JavaScript изначально используется для написания почти всего кода SPA, а не добавляется в уже готовое приложение для реализации второстепенной интерактивности, чрезвычайно важно понимать, как устроен этот язык. Обсуждаются переменные, синтаксис и функции, а также более сложные темы: контекст выполнения, замыкания и прототипы объектов.

В главе 3 дается введение в архитектуру SPA, используемую в этой книге. Здесь же читатель знакомится с основным модулем пользо-

вательского интерфейса – Shell. Модуль Shell координирует работу специализированных модулей и событий браузера, а также содержит средства для работы с данными, в частности URL-адресами и куками. Демонстрируются реализация обработчика событий и использование паттерна якорного интерфейса для управления состоянием страницы.

В главе 4 детально рассматриваются функциональные модули, которые предоставляют SPA четко определенные возможности, ограниченные областью видимости. Проводится сравнение правильно написанных функциональных модулей со сторонним JavaScript-кодом. Подчеркивается важность изоляции для обеспечения качества и модульности кода.

В главе 5 описывается построение модуля Model, который консолидирует всю бизнес-логику в едином пространстве имен. Этот модуль изолирует своих клиентов от управления данными и взаимодействия с сервером. Здесь же проектируется и разрабатывается People API. Производится тестирование модели с помощью модуля подставных данных Fake и консоли JavaScript.

В главе 6 завершается работа над модулем Model. Проектируется и разрабатывается Chat API, который также тестируется с помощью модуля Fake и консоли JavaScript. Появляется модуль Data, и приложение модифицируется для работы с «живыми» данными от веб-сервера.

В главе 7 читатель знакомится с веб-сервером Node.js. Поскольку большая часть кода SPA находится на стороне клиента, серверная часть может быть написана на любом языке, достаточно производительном, чтобы справиться с запросами приложения. Написание серверной части на JavaScript обеспечивает единообразие сред программирования и упрощает разработку всей системы. Для тех, кто незнаком с Node.js, эта глава послужит отличным введением, а опытные пользователи Node.js узнают из нее о роли сервера в SPA.

В главе 8 мы продолжаем спускаться по стеку компонентов и переходим к базе данных. Мы используем MongoDB, потому что это проверенная на практике промышленная база данных, в которой данные хранятся в виде документов в формате JSON – том самом, который применяется для передачи данных клиентам. Прежде чем переходить к рассмотрению роли базы данных в SPA, мы приводим краткое введение в MongoDB для непосвященных.

В главе 9 обсуждаются некоторые концептуальные детали, отличающие SPA от традиционного веб-приложения в стиле MVC: оптимизация SPA для поисковых систем, сбор аналитических данных и

протоколирование ошибок. Мы также рассматриваем некоторые вопросы, интересные и для традиционных веб-приложений, но особенно важные при разработке SPA: быстрая доставка статического содержимого с помощью CDN-сетей и кэширование на всех уровнях стека.

Приложение А посвящено подробному рассмотрению применяемых нами стандартов кодирования на JavaScript; пригодятся они вам или нет, вам решать, но мы находим их чрезвычайно полезными для структурирования JavaScript-кода в одностраничном приложении, потому что они обеспечивают тестопригодность, простоту сопровождения и удобочитаемость. Мы объясняем, почему стандарты кодирования так важны, описываем способы организации и документирования кода, подход к именованию переменных и методов, защиту пространств имен, организацию файлов и использование программы JSLint для проверки JavaScript-кода.

Приложение Б посвящено тестированию SPA. На эту тему можно было бы написать отдельную книгу, но она настолько важна, что мы просто не смогли оставить ее без внимания. Мы рассматриваем настройку режима тестирования, выбор каркаса тестирования, создание комплекта тестов и учет тестирования при написании модулей SPA.

Предполагаемая аудитория

Эта книга рассчитана на веб-разработчиков, архитекторов и менеджеров продуктов, имеющих хотя бы поверхностные знания о JavaScript, HTML и CSS. Если вы никогда не занимались веб-разработкой, то эта книга не для вас, хотя мы будем рады, если вы ее все равно купите (давайте, давайте, папочке нужна новая машина). Есть немало хороших книг, в которых повествуется об азах проектирования и разработки сайтов, но эта не из их числа.

Назначение этой книги – стать полезным руководством по проектированию и построению крупномасштабных одностраничных веб-приложений (SPA), в которых на всех уровнях стека применяется JavaScript. Мы используем язык JavaScript и в базе данных, и на веб-сервере, и в браузерном приложении. Примерно две трети книги посвящено разработке клиентской части. А в оставшейся трети показано, как на JavaScript написать серверную часть, воспользовавшись Node.js и MongoDB. Если вы вынуждены работать на другой серверной платформе, то большую часть логики можно будет легко перенести, но для реализации обмена сообщениями все-таки необходим событийно-управляемый веб-сервер.

Графические выделения и загрузка исходного кода

Весь исходный код в листингах и в тексте выделяется моноширинным шрифтом. Листинги сопровождаются аннотациями, иллюстрирующими основные идеи.

Весь исходный код, представленный в этой книге, можно скачать с сайта издательства по адресу www.manning.com/SinglePageWebApplications.

Требования к программному обеспечению и оборудованию

У тех, кто работает с последними версиями Mac OS X или Linux, не должно возникнуть проблем с приведенными в книге примерами, при условии что будет установлено все упоминаемое нами программное обеспечение.

Если вы работаете с Windows, то при выполнении упражнений из частей I и II вряд ли возникнут сложности. Но в части III используются программы, которые на платформе Windows либо отсутствуют, либо имеют ограничения. Мы рекомендуем воспользоваться какой-нибудь бесплатной виртуальной машиной (см. http://www.oracle.com/technetwork/server-storage/virtualbox/downloads/index.html) и дистрибутивом Linux (мы рекомендуем Ubuntu Server 13.04, см. http://www.ubuntu.com/download/server).

Автор в сети

Приобретение книги «Одностраничные веб-приложения» открывает бесплатный доступ к закрытому форуму, организованному издательством Manning Publications, где вы можете оставить свои комментарии к книге, задать технические вопросы и получить помощь от автора и других пользователей. Получить доступ к форуму и подписаться на список рассылки можно на странице www.manning.com/SinglePageWebApplications. Там же написано, как зайти на форум после регистрации, на какую помощь можно рассчитывать, и изложены правила поведения в форуме.

Издательство Manning обязуется предоставлять читателям площадку для общения с другими читателями и автором. Однако это не

означает, что автор обязан как-то участвовать в обсуждениях; его присутствие на форуме остается чисто добровольным (и не оплачивается). Мы советуем задавать автору какие-нибудь хитроумные вопросы, чтобы его интерес к форуму не угасал!

Форум автора в сети и архивы будут доступны на сайте издательства до тех пор, пока книга не перестанет печататься.

Об авторах

Майкл С. Миковски – удостоенный наград промышленный дизайнер и архитектор SPA с 13-летним опытом работы в качестве архитектора и разработчика всех компонентов веб-приложений. Четыре года он работал руководителем разработки на высокопроизводительной и высокодоступной платформе, обслуживающей сотни миллионов запросов в день и состоящей из большого кластера серверов приложений, написанных на mod_perl.

Он начал работать над коммерческими одностраничными веб-приложениями в 2007 году, когда разрабатывал сайт «Где купить» для компании AMD, а ограничения по хостингу исключали почти все другие решения. Затем, воодушевленный возможностями, которые открывали SPA, он спроектировал и разработал еще много подобных решений. Он твердо верит, что проектирование с учетом качества, «творческое разрушение»[1], минимализм и целенаправленное тестирование могут устранить сложности и непонимание, связанные с разработкой SPA.

Майк участвует во многих проектах с открытым исходным кодом и опубликовал ряд подключаемых модулей для jQuery. Он выступал с докладами на конференциях разработчиков по HTML5 в 2012 и 2013 годах, на конференции Developer Week 2013 года в Университете Сан-Франциско и на семинарах в различных компаниях. В последнее время занимается архитектурой пользовательских интерфейсов, консультированием и инженерией пользовательского восприятия.

Джош К. Пауэлл занимается вебом еще с тех пор, когда IE 6 считался хорошим браузером. Имея 13-летний опыт работы в области программной инженерии и веб-архитектуры, он обожает разрабатывать веб-приложения и организовывать команды для этого дела. В настоящее время увлечен экспериментами с различными технологиями

[1] См. http://ru.wikipedia.org/wiki/Креативное_разрушение. – *Прим. перев.*

построения одностраничных веб-приложений и отдает этому все свое время.

По странной прихоти природы, он получает заряд энергии от выступлений на публике. Он проводил презентации, посвященные одностраничным приложениям и JavaScript на различных конференциях, в том числе HTML 5 Developers Conference и NoSQL Now!, в университетах и в таких компаниях из Кремниевой долины, как Engine Yard, RocketFuel, и многих других. Он также пишет статьи для сайта www.learningjquery.com и различных сетевых журналов.

Об иллюстрации на обложке

Рисунок на обложке книги называется «Gobenador de la Abisinia», или Губернатор Абиссинии, нынешней Эфиопии. Он взят из испанского собрания местных костюмов, впервые изданного в Мадриде в 1799 году. На титульном листе книги имеется следующая надпись:

> *Coleccion general de los Trages que usan actualmente todas las Nacionas del Mundo desubierto, dibujados y grabados con la mayor exactitud por R.M.V.A.R. Obra muy util y en special para los que tienen la del viajero universal.*

В переводе это означает:

> *Всеобщая коллекция костюмов, ныне используемых народами известного мира, подготовленная и напечатанная со всей возможной точностью R. M. V. A. R. Этот труд особенно полезен тем, кто считает себя путешественником по миру.*

Хотя ничего не известно о художниках, граверах и рабочих, которые вручную раскрашивали этот рисунок, бросается в глаза, что он действительно выполнен «со всей возможной точностью». «Gobenador de la Abisinia» – лишь один из множества рисунков в этой цветной коллекции. Их разнообразие – красноречивое свидетельство неповторимости и самобытности костюмов, которые носили народы разных стран 200 лет назад.

Издательство Manning откликается на новации и инициативы в компьютерной отрасли обложками своих книг, на которых представлено широкое разнообразие местных укладов быта в позапрошлом веке. Мы возвращаем его в том виде, в каком оно запечатлено на рисунках из этого собрания.

Часть I

Введение в SPA

За то время, что вам требуется для чтения этой страницы, будет потрачено 35 млн человеко-минут на ожидание загрузки традиционных веб-страниц. Этого достаточно, чтобы 96 раз доставить марсоход Curiosity на Марс и обратно. Отношение издержек к производительности, характерное для традиционных сайтов, изумляет, и для бизнеса это может иметь самые печальные последствия. Наткнувшись на медленный сайт, пользователь уйдет – прямо к гостеприимно распахнутым кошелькам конкурентов.

Одна из причин медленной работы традиционных сайтов заключается в том, что популярные серверные каркасы для создания MVC-приложений «заточены» под передачу статических страниц «тупому» по сути своей клиенту. Например, когда мы щелкаем по ссылке на традиционном сайте, демонстрирующем слайд-шоу, экран мигает, и в течение нескольких секунд происходит перезагрузка всей страницы целиком: элементов навигации, рекламных баннеров, верхнего и нижнего колонтитулов и текста. Хотя изменилась только текущая фотография и, быть может, пояснительный текст. Хуже того, нет никакого индикатора, показывающего, что некий элемент на странице снова готов к взаимодействию с пользователем. Например, иногда по ссылке можно щелкнуть сразу после ее появления на странице, а иногда приходится ждать, пока вся страница перерисуется, а потом еще пяток секунд. Такое медленное, непоследовательное и неэффективное функционирование становится неприемлемым для все более требовательных пользователей веб.

Приготовьтесь к знакомству с другим – и, смеем сказать, лучшим – подходом к разработке веб-приложений – одностраничными приложениями (SPA). SPA переносят в браузер приемы работы, привычные для персональных приложений. В результате мы получаем быструю реакцию, которая не раздражает, а, наоборот, удивляет и восхищает пользователей. В первой части книги мы рассмотрим следующие вопросы.

- Что такое SPA и какие у него преимущества, по сравнению с традиционными сайтами?

- Как подход на основе SPA позволяет резко сократить время реакции приложения и сделать его притягательным?
- Как отточить навыки работы с JavaScript, необходимые для разработки SPA?
- Как создать демонстрационное SPA?

Дизайн продукта все чаще становится решающим фактором, определяющим успех или провал коммерческого, да и корпоративного веб-приложения. Одностраничные приложения нередко обеспечивают оптимальный, с точки зрения пользователя, способ работы. Поэтому мы ожидаем, что спрос на дизайн с упором на пользователя станет причиной более широкого распространения и большей изощренности SPA.