
Краткое содержание

Предисловие	16
Введение	18
Глава 1. Среда разработки	22
Глава 2. Введение в API	31
Глава 3. Создание веб-приложения с помощью Node и Express	35
Глава 4. Наш первый GraphQL API	39
Глава 5. База данных	53
Глава 6. Операции CRUD	66
Глава 7. Учетные записи пользователей и аутентификация	75
Глава 8. Действия пользователя	88
Глава 9. Детали	103
Глава 10. Развертывание API	110
Глава 11. Интерфейсы пользователей и React	118
Глава 12. Построение веб-клиента с помощью React	126
Глава 13. Стилизовое оформление приложения	138
Глава 14. Работа с Apollo Client	151
Глава 15. Аутентификация и состояние	169
Глава 16. Операции создания, чтения, изменения и удаления	193
Глава 17. Развертывание приложения	216

Глава 18. Создание десктопных приложений с помощью Electron.....	222
Глава 19. Интеграция веб-приложения в Electron	229
Глава 20. Развертывание Electron	237
Глава 21. Мобильные приложения на React Native	242
Глава 22. Оболочка мобильного приложения.....	248
Глава 23. GraphQL и React Native.....	264
Глава 24. Аутентификация в мобильном приложении	281
Глава 25. Дистрибуция мобильного приложения.....	307
Послесловие	316
Приложение А. Локальное выполнение API.....	317
Приложение Б. Локальное выполнение веб-приложения	318
Об авторе	319
Об обложке	320

Оглавление

Предисловие	16
Введение	18
Для кого эта книга	18
Структура	18
Условные обозначения.....	19
Использование примеров кода.....	20
Благодарности	20
От издательства.....	21
Глава 1. Среда разработки	22
Текстовый редактор	23
Терминал	23
Использование отдельного приложения терминала	23
Использование VSCode.....	23
Навигация по файловой системе	24
Инструменты командной строки и Homebrew (только для Mac)	25
Node.js и NPM	25
Установка Node.js и NPM для macOS.....	26
Установка Node.js и NPM для Windows	26
MongoDB	26
Установка и запуск MongoDB для macOS.....	27
Установка и запуск MongoDB для Windows.....	27
Git.....	28
Expo.....	28
Prettier	29
ESLint	29

Наводим красоту	30
Итоги	30
Глава 2. Введение в API	31
Что мы создаем.....	31
Как мы будем это создавать.....	32
Начало	32
Итоги	34
Глава 3. Создание веб-приложения с помощью Node и Express	35
Hello World	35
Nodemon	36
Расширение опций порта	37
Итоги	38
Глава 4. Наш первый GraphQL API.....	39
Превращение сервера в API (ну, вроде того).....	39
Основы GraphQL	43
Схемы	43
Распознаватели	44
Адаптация API	45
Итоги	52
Глава 5. База данных	53
Начало работы с MongoDB	54
Подключение MongoDB к приложению	56
Чтение и запись данных	59
Итоги	65
Глава 6. Операции CRUD	66
Разделение GraphQL-схемы и распознавателей.....	66
Написание CRUD-схемы	70
CRUD-распознаватели.....	70
Время и дата.....	72
Итоги	74
Глава 7. Учетные записи пользователей и аутентификация	75
Процесс аутентификации в приложении	75

Шифрование и токены	76
Шифрование паролей	76
JSON Web Token	78
Интеграция аутентификации в API	79
Пользовательские схемы	79
Распознаватели аутентификации	81
Добавление пользователя в контекст распознавателя	84
Итоги	87
Глава 8. Действия пользователя	88
Подготовка	88
Прикрепление пользователя к новым заметкам	89
Пользовательские разрешения на изменение и удаление	91
Запросы пользователей	92
Избранные заметки	95
Вложенные запросы	100
Итоги	102
Глава 9. Детали	103
Передовые методы Express.js для веб-приложений	103
Express Helmet	103
Совместное использование ресурсов между источниками	104
Пагинация	104
Ограничения данных	107
Прочие соображения	108
Тестирование	108
Подписки	108
Платформа Apollo GraphQL	109
Итоги	109
Глава 10. Развертывание API	110
Размещение базы данных	110
Развертывание приложения	114
Настройка проекта	115
Развертывание	117
Тестирование	117
Итоги	117

Глава 11. Интерфейсы пользователей и React	118
JavaScript и UI	119
Декларативные интерфейсы в JavaScript	120
Достаточно одного React	120
Итоги	125
Глава 12. Построение веб-клиента с помощью React	126
Что мы создаем.....	126
Как мы будем это создавать.....	127
Начало	128
Создание приложения	129
Маршрутизация	131
Создание ссылок.....	134
Компоненты UI.....	135
Итоги	137
Глава 13. Стилизовое оформление приложения	138
Создание компонента макета	138
CSS.....	140
CSS-in-JS	141
Создание компонента Button.....	142
Добавление глобальных стилей	144
Стили компонентов	146
Итоги	150
Глава 14. Работа с Apollo Client	151
Настройка Apollo Client	152
Запросы к API.....	154
Стиль.....	161
Динамические запросы	163
Пагинация.....	166
Итоги	168
Глава 15. Аутентификация и состояние	169
Создание формы регистрации.....	169
Формы и состояния в React.....	173
Мутация signUp	174
JSON Web Token и локальное хранилище	177

Переадресация.....	178
Прикрепление заголовков к запросам	178
Управление локальным состоянием	179
Выход из системы	182
Создание формы авторизации.....	185
Защищенные маршруты.....	190
Итоги	192
Глава 16. Операции создания, чтения, изменения и удаления	193
Создание заметок.....	193
Чтение заметок пользователей.....	199
Изменение заметок	202
Удаление заметок	209
Добавление в «Избранное»	211
Итоги	215
Глава 17. Развертывание приложения.....	216
Статические сайты	216
Конвейер развертывания.....	217
Хостинг исходного кода с помощью Git.....	218
Развертывание с помощью Netlify.....	219
Итоги	221
Глава 18. Создание десктопных приложений с помощью Electron	222
Что мы создаем.....	222
Как мы будем это создавать.....	223
Начало	223
Наше первое приложение на Electron.....	224
Детали окна приложения для macOS.....	226
Инструменты разработчика	226
Electron API	227
Итоги	228
Глава 19. Интеграция веб-приложения в Electron.....	229
Интеграция веб-приложения.....	229
Предупреждения и ошибки.....	230
Конфигурация	232

Политика защиты контента CSP	233
Итоги	236
Глава 20. Развертывание Electron	237
Electron Builder.....	237
Настройка Electron Builder	238
Сборка для нашей текущей платформы	239
Иконки приложения	239
Сборка для нескольких платформ.....	240
Подписание кода	240
Итоги	241
Глава 21. Мобильные приложения на React Native.....	242
Что мы создаем.....	243
Как мы будем это создавать.....	243
Начало	244
Итоги	247
Глава 22. Оболочка мобильного приложения	248
Из чего состоит React Native.....	248
Style и Styled Components	250
Styled Components	252
Маршрутизация	254
Маршрутизация по вкладкам с помощью React Navigation	254
Навигация по стеку	257
Добавление заголовков экранам.....	261
Иконки	261
Итоги	263
Глава 23. GraphQL и React Native	264
Создание списка и прокручиваемого содержимого	264
Маршрутизация списка	270
GraphQL с Apollo Client	272
Написание GraphQL-запросов	273
Добавление индикатора загрузки	279
Итоги	280

Глава 24. Аутентификация в мобильном приложении	281
Поток аутентификации	281
Создание формы авторизации.....	289
Аутентификация с помощью GraphQL-мутаций.....	293
Аутентифицированные GraphQL-запросы.....	296
Добавление формы регистрации	299
Итоги	306
Глава 25. Дистрибуция мобильного приложения.....	307
Настройка app.json	307
Иконки и экраны загрузки приложения.....	309
Иконки приложения.....	310
Экраны-заставки.....	310
Публикация через Expo	311
Создание нативных сборок	312
iOS.....	313
Android	314
Дистрибуция через магазины приложений.....	315
Итоги	315
Послесловие	316
Приложение А. Локальное выполнение API	317
Приложение Б. Локальное выполнение веб-приложения	318
Об авторе	319
Об обложке	320

«Разработка на JavaScript» — это невероятная книга, которая даст все, что вам нужно, для создания приложений с помощью JavaScript на любой платформе. Сейчас JavaScript действительно встречается буквально везде, и в этой книге был проделан уникальный труд по совмещению информации для разработчиков всех уровней. Прочтите ее, напишите код и уверенно принимайте технологические решения.

*Ева Порцелло (Eve Porcello),
разработчик ПО и инструктор в Moon Highway*

«Разработка на JavaScript» — это идеальный спутник для путешествия по не-престанно меняющейся экосистеме современного JavaScript. В этой книге Адам простым и понятным способом учит использовать React, React Native и GraphQL для разработки надежных веб-, мобильных и десктопных приложений.

*Пегги Райзис (Peggy Rayzis),
главный инженер в Apollo GraphQL*

*Посвящается моему отцу, который собрал со мной
мой первый компьютер и вычитал каждую
мою статью. Меня бы не было здесь без тебя.
Мне тебя не хватает.*

Предисловие

В 1997-м мне оставалось два года до выпуска из школы. Однажды мы с другом возились с подключенным к интернету компьютером в нашей школьной библиотеке, и он показал мне, что можно щелкнуть **View -> Source** и просмотреть основной код веб-страницы. Несколько дней спустя уже другой товарищ показал мне, как публиковать собственный HTML. Мой мозг просто взорвался.

С этого момента я подсел. Я начал собирать крупницы информации с сайтов, которые мне нравились, постепенно создавая из них собственный Франкен-сайт. Большую часть свободного времени я возился за собранным по частям компьютером, стоявшим у нас в столовой. Я даже написал (честно говоря, скопировал и вставил) свой первый JS-код, чтобы реализовать стили наведения на ссылки, что на тот момент еще нельзя было сделать с помощью простого CSS.

И вследствие поворота событий, который я рассматриваю как благоприятную версию фильма «Почти знаменит», мой доморощенный музыкальный сайт стал весьма популярен. Благодаря этому я начал получать по почте промо-CD и стал званым гостем на различных концертах. И все же самым важным для меня было то, что я мог общаться с людьми по всему миру. Я был скучающим подростком из глубинки, который любил музыку и при этом мог общаться с людьми, которых никогда не встретит в живую. Это ощущение как тогда, так и сейчас очень вдохновляет.

Сегодня мы можем создавать мощные приложения с помощью одних только веб-технологий, но мысль о том, чтобы приступить к этому, может пугать. API являются невидимым фоном, предоставляющим данные, **View -> Source** показывает конкатенированный и минифицированный код, аутентификация и безопасность таинственны, а совмещение всего этого воедино может показаться чем-то и вовсе невыполнимым. Если же взглянуть на то, что кроется за всей этой неразберихой, то можно заметить, что с помощью тех же технологий, которые я использовал еще 20 лет назад, сегодня можно не только создавать мощные веб-, десктопные или нативные мобильные приложения, но также проектировать 3D-анимацию и даже программировать роботов.

Занимаясь преподаванием, я обнаружил, что многие из нас лучше всего учатся, создавая нечто новое, разбирая это и адаптируя к конкретным случаям использования. Эту же цель преследует и моя книга. Если вы немного знакомы с HTML, CSS и JavaScript, но навыков объединить все это для создания

надежных придуманных вами приложений у вас пока маловато, то эта книга для вас. Я познакомлю вас с созданием API, который может применяться для пользовательских интерфейсов веб-, десктопных и нативных мобильных приложений. Но важнее всего то, что вы получите понимание принципов совместимости всех этих частей, которое позволит вам разрабатывать и создавать прекрасные программы.

Мне не терпится увидеть, что у вас получится.

Адам

Введение

Идея этой книги возникла у меня после написания первого десктопного приложения на Electron. Сделав карьеру веб-разработчика, я был захвачен возможностями использования веб-технологий для создания кроссплатформенных приложений. В то же время происходил рост популярности React, React Native и GraphQL. Я искал ресурсы, которые бы помогли мне научиться совмещать все эти компоненты, но ничего не нашел. Так что эта книга представляет собой руководство, которого мне тогда очень не хватало.

Ее конечная цель — научить вас использовать возможности одного только JavaScript для создания всех видов приложений.

Для кого эта книга

Она предназначена для разработчиков среднего уровня, имеющих некоторый опыт работы с HTML, CSS и JS, а также для амбициозных новичков, желающих изучить инструменты, необходимые для запуска проектов для бизнеса и любых других целей.

Структура

Структура книги подразумевает поэтапную разработку примера приложения для различных платформ. По сути, она делится на следующие разделы:

- глава 1 знакомит с настройкой среды разработки JavaScript;
- главы 2–10 посвящены созданию API с использованием Node, Express, MongoDB и Apollo Server;
- в главах 11–25 рассматриваются детали создания кроссплатформенных пользовательских интерфейсов с помощью React, Apollo и ряда других инструментов. А именно:
 - глава 11 знакомит с принципами разработки пользовательского интерфейса и рассказывает о React;

- главы 12–17 демонстрируют создание веб-приложения при помощи React, Apollo Client и CSS-in-JS;
- главы 18–20 поэтапно поясняют создание простых приложений на Electron;
- главы 21–25 учат использовать React Native и Expo для разработки мобильных приложений под iOS и Android.

Условные обозначения

В данной книге используются следующие типографские обозначения:

Курсив

Обозначает новые термины.

Интерфейс

Обозначает URL, адреса электронной почты, названия файлов и их расширения.

Моноширинный шрифт

Используется для примеров программ, а также внутри абзацев для ссылки на элементы программ, такие как переменные или имена функций, базы данных, переменные среды, операторы и ключевые слова.

Моноширинный курсив

Текст, который следует заменить значениями, введенными пользователем, или значениями, определяемыми контекстом.



Этот элемент означает подсказку или предложение.



Этот элемент обозначает общее примечание.



Этот элемент указывает на предупреждение или предостережение.

Использование примеров кода

Сопроводительный материал (примеры кода, упражнения и т. д.) доступен для загрузки по ссылке <https://github.com/javascripteverywhere>.

Цель этой книги — помочь вам справиться с задачей. В целом предложенные в ней примеры кода вы можете без проблем использовать в собственных программах и документации. Для этого не нужно спрашивать у нас разрешения, если только вы не соберетесь использовать существенную его часть. Например, написание программы, использующей несколько кусков кода из этой книги, не требует разрешения, а вот для продажи или распространения примеров из книг издательства O'Reilly разрешение уже требуется. Ответы на вопросы, использующие цитирование книги или кода, не требуют разрешения. Для включения же существенного количества примеров кода в документацию вашего продукта разрешение уже нужно.

Благодарности

Благодарю всех замечательных сотрудников O'Reilly, как бывших, так и нынешних, которые на протяжении нескольких лет были открыты и шли навстречу моим идеям. Отдельно хочу поблагодарить своего редактора Анджелу Руфино (Angela Rufino), которая давала отзывы, приободряла и делала много полезных напоминаний. Хочу также сказать спасибо Майку Лукидесу (Mike Loukides), который поддерживал меня не только кофеином, но и дружеской беседой. Выражаю признательность Дженнифер Поллок (Jennifer Pollock) за ее поддержку и воодушевление.

Я также бесконечно благодарен сообществу open source за предоставленные мне знания, которые я смог с успехом применить. Без отдельных людей и организаций, разработавших и обслуживающих множество упоминаемых мной библиотек, написание этой книги стало бы невозможным.

Научные редакторы помогли мне улучшить ее, обеспечив точность использования всех элементов. Спасибо вам, Энди Нгом (Andy Ngom), Брайан Слеттен (Brian Sletten), Максимилиано Фиртман (Maximiliano Firtman) и Зишан Чавдхари (Zeeshan Chawdhary). Вы сделали грандиозную работу по проверке кода, за что я искренне благодарен. Отдельное спасибо отправляется в адрес моего давнего коллеги и друга Джимми Уилсона (Jimmy Wilson), которому я позвонил уже в одиннадцатом часу, чтобы он просмотрел весь материал и дал отзыв. Это была существенная просьба, но он по своему обыкновению отреагировал на нее с энтузиазмом. Должен отметить, что без его помощи эта книга не появилась бы в ее нынешнем виде.

На протяжении всей взрослой жизни мне невероятно везло с умными, увлеченными и поддерживающими меня коллегами. Я получил много технических

и жизненных уроков благодаря общению с этими людьми. Перечислить всех их не представляется возможным, но я хочу выразить признательность Элизабет Бонд (Elizabeth Bond), Джону Полу Догину (John Paul Doguin), Марку Эшеру (Marc Esher), Дженн Ласситер (Jenn Lassiter) и Джессике Шафер (Jessica Schafer).

Весь процесс написания сопровождался музыкой, и эта книга могла не увидеть свет без прекрасных композиций, созданных Чаком Джонсоном (Chuck Johnson), Мэри Латтимор (Mary Lattimore), Макайей Маккрэвен (Makaya McCraven), Г. С. Шрейем (G. S. Schray), Сэмом Уилкесом (Sam Wilkes), Хироши Йошимурой (Hiroshi Yoshimura) и многими другими.

В заключение я хочу поблагодарить свою жену Эбби (Abbey) и наших детей Райли (Riley), Харрисона (Harrison) и Харлоу (Harlow), которым я зачастую не мог уделить достаточно внимания, так как был занят написанием книги. Спасибо, что терпели мои постоянные задержки в офисе и то, что даже будучи с вами, я не мог перестать думать о работе. Вы четверо являетесь мощнейшей мотивацией для всего, что я делаю.

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На веб-сайте издательства www.piter.com вы найдете подробную информацию о наших книгах.

ГЛАВА 1

Среда разработки

Джон Вуден (John Wooden), покойный тренер баскетбольной команды UCLA, считается одним из наиболее успешных тренеров всех времен. Его команда в течение двенадцати лет выиграла 10 национальных чемпионатов. В ее состав входили лучшие игроки, включая таких членов зала славы, как Лью Алсиндор (Lew Alcindor) (настоящее имя — Карим Абдул-Джаббар [Kareem Abdul-Jabbar]) и Билл Волтон (Bill Walton). Вуден имел обыкновение в первый день тренировки усаживать всех только что нанятых игроков — лучших в США среди всех старших школ — и учить их правильно надевать носки. В ответ на вопросы по этому поводу Вуден заявлял (<https://oreil.ly/lnZkf>), что «именно из мелочей складываются большие дела».

Шеф-повара описывают подготовку необходимых для приготовления блюд инструментов и ингредиентов выражением *mise en place*, что означает «все на своих местах». Благодаря учету всех мелких деталей такая подготовка позволяет им успешно справляться с готовкой во время повышенной загруженности. По аналогии с баскетбольным тренером, подготавливающим своих игроков, и поварами, готовящими блюда к обеденному наплыву клиентов, нам тоже стоит уделить время настройке среды разработки.

Для создания ее эффективного варианта вовсе не обязательно использовать дорогое ПО или самое производительное оборудование. Вообще я призываю вас начать с простого — задействовать открытое программное обеспечение (OSS) и постепенно добавлять к нему нужные инструменты. Хотя бегун и предпочитает определенный бренд кроссовок, а плотник может выбирать особенный молоток, для выработки таких предпочтений требуется время и опыт. Экспериментируйте с инструментами, наблюдайте за появлением новых и со временем вы создадите максимально удобную для вас среду разработки.

В этой главе мы сосредоточимся на установке текстового редактора, Node.js, Git, MongoDB и нескольких полезных JS-пакетов, а также определим место для терминального приложения. Возможно, у вас уже есть удобно настроенная среда разработки. Но мы тем не менее также установим несколько необходимых инструментов, которые будем использовать на протяжении всей книги. Даже

если вы, как и я, предпочитаете пропускать инструкции, то в данном случае я все же советую с ними ознакомиться.

Если в какой-то момент у вас возникнут сложности с пониманием материала книги, смело обращайтесь к ее сообществу через канал Spectrum по адресу spectrum.chat/jseverywhere.

Текстовый редактор

Текстовые редакторы во многом похожи на одежду: они нам необходимы, но личные предпочтения при этом могут сильно различаться. Некоторым нравятся простые и грамотно организованные капсулы, другие предпочитают яркий принт пейсли. Неверных решений тут не бывает, и вам следует использовать наиболее удобный для вас вариант.

Если у вас все еще нет фаворита, я очень рекомендую Visual Studio Code (<https://code.visualstudio.com>). Этот редактор с открытым исходным кодом доступен для платформ Mac, Windows и Linux. Он предлагает встроенные функции для упрощения разработки и легко изменяется при помощи расширений, предлагаемых сообществом. Кстати говоря, разработан он также на JavaScript.

Терминал

Если вы используете VSCode, то у него есть встроенный терминал. Для большинства задач по разработке этого вполне достаточно. Лично я предпочитаю использовать отдельный терминальный клиент, поскольку так мне проще управлять несколькими вкладками и использовать больше выделенного оконного пространства на компьютере. Предлагаю вам попробовать оба варианта и выбрать наиболее оптимальный.

Использование отдельного приложения терминала

Все операционные системы снабжены встроенным приложением терминала, который послужит отличной отправной точкой. В macOS он называется вполне себе очевидно: Terminal. В ОС Windows, начиная с Windows 7, аналогичная программа называется PowerShell. В дистрибутивах Linux название терминала может быть разным, но зачастую содержит слово Terminal.

Использование VSCode

Для получения доступа к терминалу в VSCode кликните **Terminal** → **New Terminal**, и перед вами откроется его окно. Командная строка будет располагаться в каталоге текущего проекта.

Навигация по файловой системе

После обнаружения терминала вам пригодится умение ориентироваться в файловой системе. Это можно делать при помощи команды `cd`, которая отвечает за смену каталога.



ИНСТРУКЦИИ КОМАНДНОЙ СТРОКИ

Инструкции терминала часто включают символы `$` или `>` в начале строки. Они используются только для указания на инструкцию, и копировать их не нужно. В данной книге я буду обозначать инструкции терминала знаком доллара (`$`). При их вводе в собственном терминале этот знак включать не надо.

Открыв приложение терминала, вы увидите командную строку с курсором, в которой можно вводить команды. По умолчанию вы находитесь в домашнем каталоге компьютера. Если вы еще этого не делали, то я рекомендую создать каталог `Projects`, который будет подкаталогом домашней директории. Этот каталог будет содержать все ваши проекты. Создать и перейти в него вы можете следующим образом:

```
# сначала вводите команду cd, которая обеспечит переход в корневую директорию:
$ cd
# затем, если вы еще не создавали каталог Projects, можете это сделать;
# для этого используйте следующую команду – она создаст Projects в виде
# подкаталога в корневой директории системы:
$ mkdir Projects
# наконец, вы можете войти в этот каталог:
$ cd Projects
```

В дальнейшем переходить в директорию `Projects` можно так:

```
$ cd # выполняет переход в корневую директорию
$ cd Projects
```

Теперь давайте предположим, что в директории `Projects` у вас есть каталог `jseverywhere`. Для перехода в него из `Projects` нужно ввести `cd jseverywhere`. Для возвращения на один каталог выше (в данном случае в `Projects`) нужно набрать `cd ..` (команда `cd`, за которой следуют две точки).

Все вместе это будет выглядеть так:

```
> $ cd # убедитесь, что вы в корневом каталоге
> $ cd Projects # для перехода из корневой директории в директорию Projects
/Projects > $ cd jseverywhere # для перехода из Projects в директорию
jsewehre
/Projects/jseverywhere > $ cd .. # для возвращения из jseverywhere в Projects
/Projects > $ # Командная строка находится в директории Projects
```

Если вы с такими вещами раньше не сталкивались, то потренируйтесь немного в перемещении между файлами, чтобы чувствовать себя более уверенно. Я выяс-