



RUBY ON RAILS ДЛЯ НАЧИНАЮЩИХ

ИЗУЧАЕМ РАЗРАБОТКУ
ВЕБ-ПРИЛОЖЕНИЙ
НА ОСНОВЕ RAILS

Майкл Хартл

УДК 004.438Ruby on Rails
ББК 32.973.22
X22

Хартл М.

X22 Ruby on Rails для начинающих / пер. с англ. А. Разуваева. – М.: ДМК Пресс, 2017. – 572 с.: ил.

ISBN 978-5-97060-429-8

Ruby on Rails – один из наиболее популярных фреймворков для разработки веб-приложений, но его изучение и использование – не самая простая задача. Эта книга поможет вам решить ее независимо от того, имеете ли вы опыт веб-разработки вообще и Rails в частности. Известный автор и ведущий разработчик Rails Майкл Хартл познакомит вас с Rails на примере разработки трех приложений. Автор рассказывает не только о Rails, но также описывает основы Ruby, HTML, CSS и SQL, которые пригодятся вам при разработке своих веб-приложений. Начиная обсуждение каждой новой методики, Хартл доходчиво объясняет, как она помогает решать практические задачи, а затем демонстрирует ее применение в программном коде, достаточно простом и понятном.

Издание предназначено для всех программистов, желающих изучить Ruby on Rails.

УДК 004.438Ruby on Rails
ББК 32.973.22

Authorized translation from the English language edition, entitled Ruby on Rails Tutorial. Learn Web Development with Rails. 3rd Edition; ISBN 9780134077703; by Michael Hartl; published by Pearson Education, Inc., publishing as Addison-Wesley Professional. Copyright © 2015 by Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc. RUSSIAN language edition published by ДМК PUBLISHERS. Copyright © 2016.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Материал, изложенный в данной книге, многократно проверен. Но поскольку вероятность технических ошибок все равно существует, издательство не может гарантировать абсолютную точность и правильность приводимых сведений. В связи с этим издательство не несет ответственности за возможные ошибки, связанные с использованием книги.

ISBN 978-0-13-407770-3 (англ.)
ISBN 978-5-97060-429-8 (рус.)

Copyright © 2015 Michael Hartl
© Перевод, оформление, издание, ДМК Пресс, 2017

Содержание

Предисловие	12
Благодарности	13
Об авторе	14
Авторские права и лицензия	15
Глава 1. От нуля к развертыванию	16
1.1. Введение	18
1.1.1. Необходимая квалификация	19
1.1.2. Типографские соглашения.....	20
1.2. За работу.....	22
1.2.1. Среда разработки	23
1.2.2. Установка Rails.....	25
1.3. Первое приложение.....	26
1.3.1. Компоновщик.....	30
1.3.2. Сервер Rails.....	33
1.3.3. Модель-представление-контроллер (MVC).....	35
1.3.4. Hello, world!.....	38
1.4. Управление версиями с Git.....	39
1.4.1. Установка и настройка.....	41
1.4.2. Что дает использование репозитория Git?	42
1.4.3. Bitbucket.....	43
1.4.4. Ветвление, редактирование, фиксация, слияние	47
1.5. Развертывание.....	50
1.5.1. Установка Heroku.....	52
1.5.2. Развертывание на Heroku, шаг первый	53
1.5.3. Развертывание на Heroku, шаг второй	54
1.5.4. Команды Heroku.....	54
1.6. Заключение.....	55
1.6.1. Что мы узнали в этой главе	55
1.7. Упражнения.....	56
Глава 2. Мини-приложение.....	58
2.1. Проектирование приложения.....	58
2.1.1. Модель пользователей	61
2.1.2. Модель микросообщений.....	61
2.2. Ресурс Users	61
2.2.1. Обзор пользователей	64
2.2.2. MVC в действии.....	67

2.2.3. Недостатки ресурса Users	74
2.3. Ресурс Microposts.....	74
2.3.1. Микрообзор микросообщений	74
2.3.2. Ограничение размеров микросообщений.....	77
2.3.3. Принадлежность множества микросообщений одному пользователю.....	79
2.3.4. Иерархия наследования.....	81
2.3.5. Развертывание мини-приложения.....	83
2.4. Заключение.....	83
2.4.1. Что мы узнали в этой главе	84
2.5. Упражнения.....	85
Глава 3. В основном статические страницы	87
3.1. Установка учебного приложения	87
3.2. Статические страницы.....	90
3.2.1. Рождение статических страниц	91
3.2.2. Доработка статических страниц	97
3.3. Начало работы с тестированием	99
3.3.1. Наши первые тесты	100
3.3.2. Красный.....	102
3.3.3. Зеленый	103
3.3.4. Рефакторинг	105
3.4. Слегка динамические страницы.....	105
3.4.1. Тестирование заголовков (КРАСНЫЙ)	106
3.4.2. Добавление заголовков страниц (ЗЕЛЕНЫЙ)	108
3.4.3. Макеты и встроенный код на Ruby (рефакторинг)	110
3.4.4. Установка корневого маршрута.....	115
3.5. Заключение.....	116
3.5.1. Что мы изучили в этой главе.....	116
3.6. Упражнения.....	117
3.7. Продвинутые настройки тестирования.....	118
3.7.1. Средства составления отчетов minitest	119
3.7.2. Настраиваем backtrace	120
3.7.3. Автоматизация тестирования с помощью Guard.....	120
Глава 4. Ruby со вкусом Rails	126
4.1. Мотивация	126
4.2. Строки и методы.....	130
4.2.1. Комментарии	131
4.2.2. Строки.....	131
4.2.3. Объекты и передача сообщений.....	134
4.2.4. Определение методов	136

4.2.5. Еще раз о вспомогательной функции заголовка	137
4.3. Другие структуры данных	138
4.3.1. Массивы и диапазоны	138
4.3.2. Блоки	142
4.3.3. Хэши и символы	144
4.3.4. Еще раз о CSS	147
4.4. Ruby-классы	149
4.4.1. Конструкторы	149
4.4.2. Наследование классов	150
4.4.3. Изменение встроенных классов	153
4.4.4. Класс контроллера	154
4.4.5. Класс User	156
4.5. Заключение	158
4.5.1. Что мы узнали в этой главе	158
4.6. Упражнения	159
Глава 5. Заполнение макета	160
5.1. Добавление некоторых структур	160
5.1.1. Навигация по сайту	161
5.1.2. Bootstrap и собственные стили CSS	166
5.1.3. Частичные шаблоны	173
5.2. Sass и конвейер ресурсов	177
5.2.1. Конвейер ресурсов	177
5.2.2. Синтаксически безупречные таблицы стилей	180
5.3. Ссылки в макете	186
5.3.1. Страница Contact	187
5.3.2. Маршруты в Rails	188
5.3.3. Использование именованных маршрутов	190
5.3.4. Тесты для проверки ссылок в макете	191
5.4. Регистрация пользователей: первый шаг	193
5.4.1. Контроллер Users	193
5.4.2. Адрес URL страницы регистрации	195
5.5. Заключение	196
5.5.1. Что мы узнали в этой главе	197
5.6. Упражнения	198
Глава 6. Моделирование пользователей	200
6.1. Модель User	201
6.1.1. Миграции базы данных	202
6.1.2. Файл модели	207
6.1.3. Создание объектов User	207

6.1.4. Поиск объектов User	210
6.1.5. Обновление объектов User	211
6.2. Проверка объектов User	212
6.2.1. Проверка допустимости	213
6.2.2. Проверка наличия	214
6.2.3. Проверка длины	216
6.2.4. Проверка формата	218
6.2.5. Проверка уникальности	222
6.3. Добавление безопасного пароля	228
6.3.1. Хэшированный пароль	229
6.3.2. Метод <code>has_secure_password</code>	231
6.3.3. Минимальная длина пароля	232
6.3.4. Создание и аутентификация пользователя	233
6.4. Заключение	235
6.4.1. Что мы узнали в этой главе	236
6.5. Упражнения	236
Глава 7. Регистрация	239
7.1. Страница профиля пользователя	239
7.1.1. Отладка и окружение Rails	241
7.1.2. Ресурс Users	245
7.1.3. Отладчик	249
7.1.4. Аватар и боковая панель	250
7.2. Форма регистрации	254
7.2.1. Применение <code>form_for</code>	256
7.2.2. Разметка HTML формы регистрации	258
7.3. Неудачная регистрация	261
7.3.1. Действующая форма	262
7.3.2. Строгие параметры	264
7.3.3. Сообщения об ошибках при регистрации	267
7.3.4. Тесты для неудачной регистрации	271
7.4. Успешная регистрация	273
7.4.1. Окончательная форма регистрации	273
7.4.2. Кратковременные сообщения	275
7.4.3. Первая регистрация	277
7.4.4. Тесты для успешной отправки формы	279
7.5. Профессиональное развертывание	280
7.5.1. Поддержка SSL	281
7.5.2. Действующий веб-сервер	282
7.5.3. Номер версии Ruby	283
7.6. Заключение	284

7.6.1. Что мы узнали в этой главе	285
7.7. Упражнения.....	285

Глава 8. Вход и выход..... 288

8.1. Сеансы.....	288
8.1.1. Контроллер Sessions.....	289
8.1.2. Форма входа.....	291
8.1.3. Поиск и аутентификация пользователя.....	294
8.1.4. Отображение кратковременных сообщений	297
8.1.5. Тест кратковременного сообщения.....	298
8.2. Вход.....	300
8.2.1. Метод log_in.....	301
8.2.2. Текущий пользователь	303
8.2.3. Изменение ссылок шаблона	306
8.2.4. Тестирование изменений в шаблоне	309
8.2.5. Вход после регистрации.....	313
8.3. Выход.....	315
8.4. Запомнить меня	317
8.4.1. Узелок на память	318
8.4.2. Вход с запоминанием.....	322
8.4.3. Забыть пользователя.....	329
8.4.4. Две тонкости	331
8.4.5. Флажок «Запомнить меня».....	334
8.4.6. Проверка запоминания	339
8.5. Заключение.....	345
8.5.1. Что мы узнали в этой главе	346
8.6. Упражнения.....	346

Глава 9. Обновление, отображение и удаление пользователей..... 350

9.1. Обновление пользователей.....	350
9.1.1. Форма редактирования.....	351
9.1.2. Неудача при редактировании	355
9.1.3. Тестирование неудачной попытки редактирования	357
9.1.4. Успешная попытка редактирования (с TDD).....	357
9.2. Авторизация.....	360
9.2.1. Требование входа пользователей.....	361
9.2.2. Требование наличия прав у пользователя.....	366
9.2.3. Дружелюбная переадресация	370
9.3. Вывод списка всех пользователей.....	374
9.3.1. Список пользователей.....	375

9.3.2. Образцы пользователей	378
9.3.3. Постраничный просмотр	380
9.3.4. Тестирование страницы со списком пользователей	382
9.3.5. Частичный рефакторинг	385
9.4. Удаление пользователей	386
9.4.1. Администраторы	386
9.4.2. Метод destroy	389
9.4.3. Тесты для проверки удаления пользователя	392
9.5. Заключение	394
9.5.1. Что мы изучили в этой главе	395
9.6. Упражнения	396

Глава 10. Активация учетной записи и сброс пароля 399

10.1. Активация учетной записи	399
10.1.1. Ресурс AccountActivations	401
10.1.2. Отправка письма для активации	406
10.1.3. Активация учетной записи	417
10.1.4. Тестирование активации и рефакторинг	424
10.2. Сброс пароля	427
10.2.1. Ресурс для сброса пароля	429
10.2.2. Контроллер и форма для сброса пароля	432
10.2.3. Метод объекта рассылки для сброса пароля	435
10.2.4. Смена пароля	441
10.2.5. Тестирование сброса пароля	447
10.3. Отправка электронных писем из эксплуатационного окружения	449
10.4. Заключение	451
10.4.1. Что мы узнали в этой главе	452
10.5. Упражнения	453
10.6. Доказательство сравнения срока годности ссылки	455

Глава 11. Микросообщения пользователей 457

11.1. Модель Micropost	457
11.1.1. Базовая модель	458
11.1.2. Проверка микросообщений	459
11.1.3. Связь User/Micropost	462
11.1.4. Усовершенствование микросообщений	464
11.2. Вывод микросообщений	468
11.2.1. Отображение микросообщений	468
11.2.2. Образцы микросообщений	472
11.2.3. Тесты профиля с микросообщениями	477
11.3. Манипулирование микросообщениями	479

11.3.1. Управление доступом к микросообщениям	480
11.3.2. Создание микросообщений.....	482
11.3.3. Прото-лента сообщений	489
11.3.4. Удаление микросообщений	494
11.3.5. Тесты микросообщений	496
11.4. Изображения в микросообщениях	499
11.4.1. Выгрузка изображений.....	499
11.4.2. Проверка изображений.....	503
11.4.3. Изменение размеров изображений.....	506
11.4.4. Выгрузка изображений в эксплуатационном окружении	508
11.5. Заключение	512
11.5.1. Что мы узнали в этой главе	513
11.6. Упражнения.....	513

Глава 12. Следование за пользователями 516

12.1. Модель Relationship	517
12.1.1. Проблема модели данных (и ее решение).....	517
12.1.2. Связь пользователь/взаимоотношения	523
12.1.3. Проверка взаимоотношений.....	525
12.1.4. Читаемые пользователи	526
12.1.5. Читатели	528
12.2. Веб-интерфейс следования за пользователями	530
12.2.1. Образцы данных.....	530
12.2.2. Статистика и форма для оформления следования	532
12.2.3. Страницы читаемых и читателей.....	540
12.2.4. Стандартная реализация кнопки «Подписаться»	547
12.2.5. Реализация кнопки «Подписаться» с применением Ajax	550
12.2.6. Тестирование подписки	553
12.3. Лента сообщений.....	555
12.3.1. Мотивация и стратегия.....	555
12.3.2. Первая реализация ленты сообщений.....	557
12.3.3. Подзапросы.....	560
12.4. Заключение	564
12.4.1. Рекомендации по дополнительным ресурсам	564
12.4.2. Что мы узнали в этой главе	565
12.5. Упражнения.....	566

Глава 1

От нуля к развертыванию

Добро пожаловать в «Ruby on Rails для начинающих: Изучаем разработку веб-приложений на основе Rails». Цель данной книги – научить вас разрабатывать собственные веб-приложения с использованием популярного фреймворка Ruby on Rails. Если эта тема для вас в новинку, «Ruby on Rails для начинающих» даст вам полное представление о разработке веб-приложений, включая основополагающие знания о Ruby, Rails, HTML и CSS, базах данных, управлении версиями, тестировании и развертывании – достаточно для начала вашей карьеры веб-разработчика или предпринимателя в сфере компьютерных технологий. С другой стороны, если вы уже знакомы с веб-разработкой, эта книга поможет вам освоить основы фреймворка Rails, включая MVC и REST, генераторы, миграции, маршрутизацию и встроенный язык Ruby. В любом случае, когда вы закончите чтение этой книги, вы будете готовы извлечь максимум пользы из более продвинутых книг, блогов и скринкастов (видеоуроков), являющихся частью обширной образовательной экосистемы¹.

В данной книге используется комплексный подход к веб-разработке: в процессе обучения мы напишем три примера приложений с возрастающей сложностью; в разделе 1.3 мы начнем создание простенького приложения, затем, в главе 2, перейдем к более продвинутому и, наконец, в главах с 3 по 12 полностью сосредоточимся на большом учебном примере. Как можно догадаться, примеры приложений в этой книге не привязаны к какой-либо категории веб-сайтов; хотя заключительный пример будет иметь значительное сходство с весьма популярной социальной сетью Twitter (которая, по совпадению, изначально была написана на Rails). Основное внимание здесь будет уделяться общим принципам разработки, поэтому полученные знания станут для вас надежным фундаментом, вне зависимости от того, какие виды приложений вы будете создавать.

Я часто слышу вопрос: «Какими знаниями нужно обладать для изучения веб-разработки с этой книгой?» Как мы увидим в разделе 1.1.1, веб-разработка – сложная тема, особенно для полных новичков. Хотя изначально книга предназначалась

¹ Самую свежую версию книги (на англ. языке. – *Прим. перев.*) можно найти на сайте <http://www.railstutorial.org/>. Если вы читаете печатную версию книги, сверьте версию своего экземпляра с онлайн-версией на <http://www.railstutorial.org/book>.

для читателей с некоторым опытом программирования, она нашла множество читателей среди начинающих разработчиков. Учитывая это, в данном (третьем) издании значительно снижен порог входа в разработку с Ruby on Rails (см. блок 1.1).

Блок 1.1 ❖ Снижение порога входа

С целью снижения порога входа для начинающих разработчиков в третьем издании сделано следующее:

- использована стандартная среда разработки, размещенная в облаке (раздел 1.2), что позволяет обойти множество проблем, связанных с установкой и настройкой новой системы;
- использован «предустановленный стек» Rails, включающий встроенный фреймворк тестирования MiniTest;
- уменьшено количество внешних зависимостей (RSpec, Cucumber, Capybara, Factory Girl);
- использован облегченный и более гибкий подход к тестированию;
- исключены сложные в настройке варианты (Spork, RubyTest);
- меньше внимания уделяется особенностям, свойственным конкретной версии Rails, и больше – общим принципам веб-разработки.

Я надеюсь, что эти изменения сделают третье издание книги доступным для еще более широкой аудитории, чем предыдущие.

В этой первой главе мы начнем изучение фреймворка Ruby on Rails с установки необходимого программного обеспечения и настройки рабочего окружения (раздел 1.2). Затем создадим первое Rails-приложение с названием `hello_app`. В книге «Ruby on Rails для начинающих» особое значение придается хорошим привычкам в программировании, поэтому сразу после создания нового Rails-проекта мы поместим его в систему управления версиями Git (раздел 1.4). И, верите вы в это или нет, в этой главе мы даже развернем наше первое приложение в Сети (раздел 1.5).

В главе 2 мы создадим второй проект для демонстрации основ работы Rails-приложения. Для быстроты в этом *мини-приложении* (с названием `toy_app`) мы применим прием «скаффолдинга» (scaffolding, см. блок 1.2) для генерации кода. Поскольку код получится одновременно и сложным, и уродливым, мы оставим его в стороне и сосредоточимся на взаимодействии с приложением через его идентификатор *URI* (который часто называют адресом *URL*)¹, с использованием веб-браузера.

Остальная часть книги описывает разработку единственного большого учебного приложения (с названием `sample_app`), весь код которого будет написан с нуля. При этом мы будем использовать фиктивные объекты, приемы разработки через тестирование (Test-Driven Development, TDD) и интеграционные тесты. В главе 3 мы сначала создадим статические страницы, а затем добавим в них немного ди-

¹ Аббревиатура URI расшифровывается как Uniform Resource Identifier (универсальный идентификатор ресурсов), а аббревиатура URL – как Uniform Resource Locator (универсальный указатель ресурсов). На практике URL обычно соответствует тому, что «находится в адресной строке браузера».

намического контента. В главе 4 мы совершим небольшое турне по языку Ruby, лежащему в основе Rails. Затем, в главах с 5 по 10, завершим базовую часть приложения, создав макет сайта, модель данных пользователя, а также систему регистрации и аутентификации (включая активацию учетной записи и сброс пароля). Наконец, в главах 11 и 12 мы добавим поддержку микроблоггинга и немного социальных функций, чтобы закончить действующий пример сайта.

Блок 1.2 ❖ Скаффолдинг: быстрее, проще, заманчивее

С самого начала фреймворк Rails привлек к себе внимание знаменитым видеороликом от Дэвида Хейнмейера Ханссона (David Heinemeier Hansson) – создателя Rails, демонстрирующим создание микроблога за 15 минут. Этот и другие подобные видеоролики – отличный способ познакомиться с возможностями Rails, и я рекомендую посмотреть их. Но предупреждаю: они совершают свой удивительный пятнадцатиминутный подвиг, используя функцию под названием «скаффолдинг» (scaffolding), которая в значительной степени опирается на код, волшебным образом сгенерированный Rails-командой `generate scaffold`.

Работая над книгой по Ruby on Rails, мне приходилось бороться с искушением положить на эту функцию, потому что это быстрее, проще, заманчивее. Но сложность и огромный объем кода, производимый генератором, могли стать непреодолимым препятствием для начинающего Rails-разработчика; вы, вероятно, сможете использовать его, но почти наверняка не сможете понять. Увлечшись скаффолдингом, вы рискуете превратиться в виртуозного генератора сценариев с весьма поверхностным знанием Rails.

В этой книге мы будем придерживаться (почти) полярно противоположного подхода: хотя в главе 2 создадим мини-приложение, сгенерировав код автоматически, основой книги все же является приложение, которое мы начнем писать в главе 3. На каждом этапе его разработки мы будем писать небольшие куски кода, достаточно простые для понимания, но все же требующие некоторых усилий для их усвоения. Результатом станет более глубокое понимание Rails, которое, в свою очередь, даст вам хорошую базу для написания практически любых типов веб-приложений.

1.1. Введение

«Ruby on Rails» (или просто «Rails») – это фреймворк для разработки веб-приложений на языке программирования Ruby. Со времен своего дебюта в 2004 году Ruby on Rails довольно быстро стал одним из самых мощных и популярных инструментов создания динамических веб-приложений. Rails используется множеством различных компаний: Airbnb, Basecamp, Disney, GitHub, Hulu, Kickstarter, Shopify, Twitter и Yellow Pages. Помимо этого, существует множество компаний, занимающихся разработкой веб-приложений и специализирующихся на Rails, таких как ENTP, thoughtbot, Pivotal Labs, Hashrocket и HappyFunCorp, плюс бесчисленное множество независимых консультантов, преподавателей и индивидуальных разработчиков.

Что же делает Rails таким замечательным? Во-первых, Ruby on Rails – это открытый исходный код, доступный на условиях лицензии MIT, и, как следствие, его можно загружать и использовать совершенно бесплатно. Rails также обязан своим успехом изящному и компактному дизайну. Используя податливость языка Ruby, лежащего в его основе, Rails фактически определяет предметно-ориентированный язык для разработки веб-приложений. В результате множество часто встречающихся задач веб-программирования, таких как динамическое создание разметки HTML, определение моделей данных и маршрутизация URL, легко решаются в Rails, а конечный код приложений получается кратким и читаемым.

Rails также быстро адаптируется к новым тенденциям в веб-технологиях. Например, Rails одним из первых полностью реализовал архитектурный стиль REST структурирования веб-приложений (мы будем изучать его на всем протяжении книги). И когда в других фреймворках появляются новые успешные приемы, создатель Rails, Дэвид Хейнмейер Ханссон (David Heinemeier Hansson), и рабочая группа Rails не стесняются использовать чужой опыт. Пожалуй, наиболее драматичным примером является слияние Rails с конкурирующим веб-фреймворком Merb, благодаря которому Rails получил модульный дизайн, стабильный API и улучшенную производительность.

Наконец, вокруг Rails сплотилось необычайно увлеченное и разнообразное сообщество: сотни разработчиков, представительные конференции, огромное количество гемов (пакетов, законченных решений конкретных задач, таких как страничный вывод и выгрузка изображений), богатый набор информативных блогов и рог изобилия форумов и IRC-каналов. Большое количество Rails-программистов также облегчает работу с (неизбежными) ошибками, возникающими в процессе разработки: алгоритм «Ищи в Google сообщение об ошибке» практически всегда помогает найти подходящую статью в блоге или сообщение на форуме.

1.1.1. Необходимая квалификация

Формально эта книга не требует некоторого набора необходимых знаний. Она содержит учебные сведения не только о фреймворке Rails, но также о лежащем в его основе языке Ruby, встроенном фреймворке тестирования (MiniTest), командной строке Unix, HTML, CSS, немного JavaScript и даже чуть-чуть SQL. Это очень объемный материал, и потому для работы с книгой желательно иметь некоторый опыт использования HTML и программирования. Однако данную книгу использовало так много новичков для изучения веб-разработки с нуля, что даже если у вас мало опыта, я все же советую попробовать. Если содержимое учебника покажется сложным, вы всегда можете сделать шаг назад и начать с одного из ресурсов, перечисленных ниже. Другая возможная стратегия (рекомендованная многими читателями): прочитать книгу дважды; возможно, вы удивитесь, как много узнали в первый раз (и насколько проще читать по второму кругу).

Стоит ли вначале изучить Ruby? Ответ зависит от вашего личного стиля обучения и опыта в программировании. Если вы предпочитаете изучать все систематически, с самых основ, или прежде никогда не программировали, вам, возмож-

но, стоит начать с изучения Ruby, и в этом случае я рекомендую книгу «Учись программировать» Криса Пайна (Chris Pine)¹ и «Начало Ruby» Питера Купера (Peter Cooper)². С другой стороны, множество начинающих Rails-разработчиков имеет своей целью создание веб-приложений и, скорее, предпочтет не корпеть над толстенной книгой, чтобы написать одну-единственную веб-страницу. В этом случае я рекомендую поработать с коротким интерактивным учебником «Try Ruby»³, чтобы получить общее представление о Ruby. Если и после этого книга окажется для вас слишком сложной, попробуйте начать с «Learn Ruby on Rails» Дэниэла Кехо (Daniel Kehoe)⁴ или «One Month Rails»⁵ – обе книги ориентированы на начинающих, в отличие от данной книги.

Независимо от того, где вы начнете, к концу этой книги вы будете готовы к чтению других, более продвинутых ресурсов по Rails. Вот некоторые из тех, которые я могу рекомендовать:

- Code School⁶: хорошие интерактивные онлайн-курсы по программированию;
- Turing School of Software & Design⁷: очные, 27-недельные курсы в Денвере, Колорадо. Читатели этой книги могут получить скидку в \$500, воспользовавшись промокодом RAILSTUTORIAL500;
- Tealef Academy⁸: хорошая образовательная онлайн-площадка, посвященная Rails-разработке (включая продвинутый материал);
- Thinkful⁹: онлайн-курс в паре с профессиональным инженером, работа по учебному плану, основанному на конкретном проекте;
- RailsCasts¹⁰ Райана Бейтса (Ryan Bates): отличные (в основном бесплатные) видеоуроки по Rails;
- RailsApps¹¹: большой выбор подробных проектов и учебников, посвященных разным темам;
- Rails Guides¹²: актуальные руководства по Rails.

1.1.2. Типографские соглашения

Соглашения в этой книге главным образом очевидны. В этом разделе я упомяну лишь те, которые таковыми не являются.

¹ <http://www.shokhirev.com/mikhail/ruby/ltp/title.html>. – *Прим. перев.*

² <http://www.amazon.com/gp/product/1430223634>.

³ <http://tryruby.org/>.

⁴ <http://learn-rails.com/learn-ruby-on-rails.html>.

⁵ <http://mbsy.co/7Zdc7>.

⁶ <https://my.getambassador.com/>.

⁷ <https://www.turing.io/friends/tutorial>.

⁸ <https://launchschool.com/>.

⁹ <https://www.thinkful.com/a/railstutorial>.

¹⁰ <http://railscasts.com/>.

¹¹ <https://tutorials.railsapps.org/>.

¹² <http://rusrails.ru/>.

В этой книге присутствует множество примеров применения командной строки. Для простоты все они используют приглашение в стиле Unix (знак доллара):

```
$ echo "hello, world"
hello, world
```

Как отмечено в разделе 1.2, я рекомендую пользователям всех операционных систем (особенно Windows) использовать облачную среду разработки (раздел 1.2.1), в которой есть встроенная командная строка Unix (Linux). Это особенно полезно, потому что в Rails имеется множество команд, которые можно запустить из командной строки. Например, в разделе 1.3.2 мы запустим локальный веб-сервер командой **rails server**:

```
$ rails server
```

По аналогии с приглашением командной строки, здесь используется соглашение о разделителях каталогов в стиле Unix (то есть прямого слеша /). Например, конфигурационный файл **production.rb** учебного приложения будет представлен как:

```
config/environments/production.rb
```

Этот путь к файлу следует интерпретировать как относительный, начинающийся в корневом каталоге приложения, абсолютный путь к которому зависит от системы; в облачной интегрированной среде разработки (IDE) (раздел 1.2.1) этот путь выглядит так:

```
/home/ubuntu/workspace/sample_app/
```

То есть полный путь к `production.rb` имеет вид:

```
/home/ubuntu/workspace/sample_app/config/environments/production.rb
```

Для простоты я часто буду опускать путь к корневому каталогу приложения и писать просто: **config/environments/production.rb**.

В книге нередко демонстрируется вывод разных программ (команд, системы управления версиями, программ на Ruby и т. д.). Из-за неисчислимого количества небольших различий между компьютерными системами вывод, который вы увидите, возможно, не всегда в точности совпадет с тем, что показан в тексте, но это не повод для беспокойства. Некоторые команды могут вызывать ошибки, связанные с особенностями вашей системы, поэтому, чтобы не решать сизифову задачу документирования всех таких погрешностей, я отсылаю вас к алгоритму «ищи сообщение об ошибке в Google», который, между прочим, является хорошей практикой для программирования. Если вы столкнетесь с проблемами в процессе обучения, я советую обратиться к ресурсам, список которых приводится в справочном разделе¹.

¹ <https://www.railstutorial.org/#help>.

Поскольку в книге (помимо всего прочего) рассматривается вопрос тестирования Rails-приложений, часто бывает полезно знать, что данный кусок кода приводит к неудаче (обозначается красным цветом) или, наоборот, к успеху тестирования (обозначается зеленым цветом). Для удобства программный код тестов будет отмечаться словами **КРАСНЫЙ** и **ЗЕЛЕНый**.

К каждой главе прилагаются упражнения, выполнение которых необязательно, но рекомендуется. Чтобы сделать основной текст независимым от упражнений, решения обычно не включаются в последующие листинги кода. В редких случаях, когда решение упражнения используется в дальнейшем, оно будет приведено в основном тексте.

Наконец, для удобства в книге применяются два соглашения, облегчающие понимание большого количества примеров кода. Во-первых, в некоторых листингах выделены одна или несколько строк, как показано ниже:

```
class User < ActiveRecord::Base
  validates :name, presence: true
  validates :email, presence: true
end
```

Такое выделение обычно указывает на наиболее важный новый код в данном примере и часто (хотя и не всегда) отражает разницу между этим и предыдущим листингом. Во-вторых, для краткости и простоты во многих листингах в книге используются вертикальные точки, например:

```
class User < ActiveRecord::Base
  .
  .
  .
  has_secure_password
end
```

Эти точки обозначают пропущенный код, и их не нужно копировать буквально.

1.2. За работу

Установка Ruby, Rails и всего сопутствующего программного обеспечения поддержки может привести в отчаяние даже опытных Rails-разработчиков. Проблема осложняется большим разнообразием окружений: разные операционные системы, версии, предпочтения в выборе текстовых редакторов и интегрированных сред разработки (IDE) и т. д. Пользователи, уже установившие среду разработки, могут сразу переходить к настройкам, а новым пользователям (как указано в блоке 1.1) я предлагаю избежать проблем с установкой и настройкой за счет использования *облачной интегрированной среды разработки*. Облачная IDE запускается в обычном браузере и, следовательно, одинаково хорошо работает на любой платформе, что особенно полезно для операционных систем (например, Windows), в которых разработка с Rails исторически была сложным занятием. Если, несмотря на все

предполагаемые трудности, вы по-прежнему хотите создать локальную среду разработки, я рекомендую следовать инструкциям на [InstallRails.com](http://installrails.com)¹.

1.2.1. Среда разработки

Принимая во внимание множество своеобразных настроек и предпочтений, число вариантов окружений разработки может равняться числу Rails-программистов. Чтобы избежать этих сложностей, я адаптировал данную книгу под превосходную облачную среду разработки Cloud9. В частности, мне было очень приятно сотрудничать с Cloud9, чтобы предложить вам среду разработки, ориентированную на потребности именно этого издания учебника. Получившееся рабочее пространство Cloud9 предоставляется предварительно настроенным, со всем программным обеспечением (включая Ruby, RubyGems, Git), необходимым для профессиональной разработки с Rails. (Отдельно мы будем устанавливать только сам фреймворк Rails, но это сделано намеренно (раздел 1.2.2).) Облачная IDE также включает три основных компонента, необходимых для разработки веб-приложений: текстовый редактор, навигатор файловой системы и терминал командной строки (рис. 1.1). Кроме того, ее текстовый редактор поддерживает глобальный поиск «Найти в файлах», который я считаю необходимым для навигации по любым большим

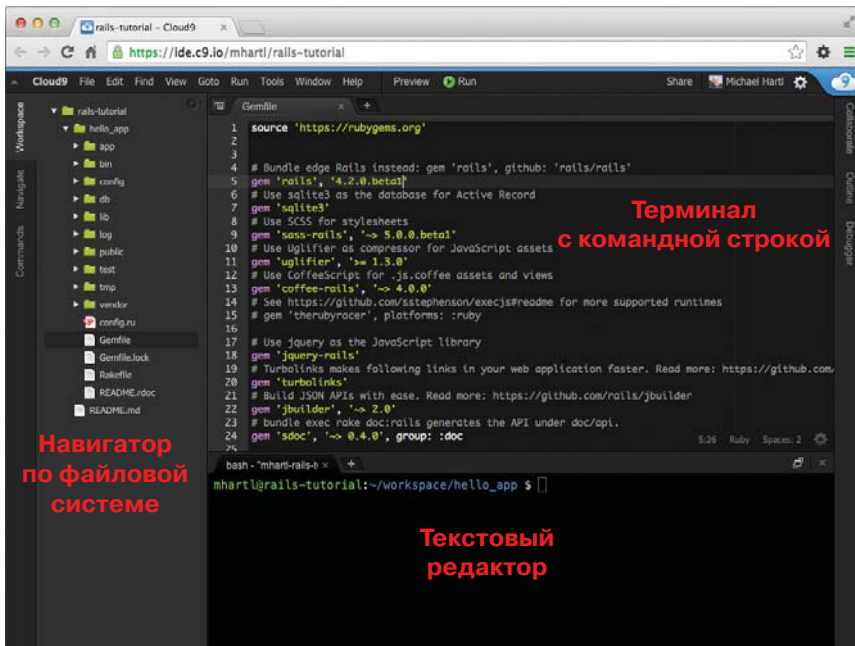


Рис. 1.1 ❖ Анатомия облачной IDE

¹ <http://installrails.com/>.

проектам Ruby или Rails¹. И наконец, если вы решите использовать не только облачную IDE (я могу лишь приветствовать изучение других инструментов), на ее примере вы получите великолепное введение в основные возможности текстовых редакторов и других инструментов разработки.

Ниже приводится пошаговая инструкция, описывающая, как начать работу с облачной средой разработки:

- создайте бесплатную учетную запись в Cloud9²;
- щелкните на ссылке **Goto your Dashboard** (Перейти к панели управления);
- щелкните на ссылке **Create New Workspace** (Создать новое рабочее пространство);
- как показано на рис. 1.2, создайте рабочее пространство rails-tutorial (не rails_tutorial), установите в настройках флажок **Private to the people I invite** (Только для приглашенных) и выберите значок **RailsTutorial** (не значок Ruby on Rails);

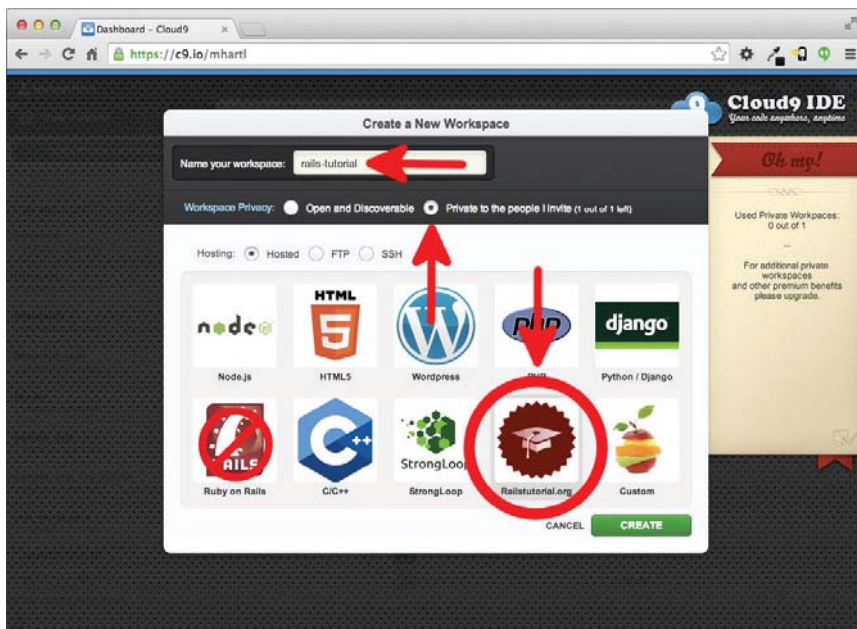


Рис. 1.2 ❖ Создание нового рабочего пространства в Cloud9

- щелкните на ссылке **Create** (Создать);
- после того как Cloud9 закончит подготовку рабочего пространства, выберите его и щелкните на ссылке **Start editing** (Начать редактирование).

¹ Например, чтобы найти определение функции `foo`, можно запустить глобальный поиск по фразе `def foo`.

² <https://c9.io/web/sign-up/free>.

Так как использование двух пробелов для отступов считается практически универсальным соглашением в Ruby, я также рекомендую изменить настройки редактора (по умолчанию используются четыре пробела). Как показано на рис. 1.3, щелкните на значке с изображением шестеренки в правом верхнем углу, выберите пункт **Code Editor (Ace)** (Редактор кода (Ace)) и измените параметр **Soft Tabs** (Мягкая табуляция). (Обратите внимание, что настройки применяются мгновенно и нет необходимости нажимать кнопку **Save** (Сохранить).)

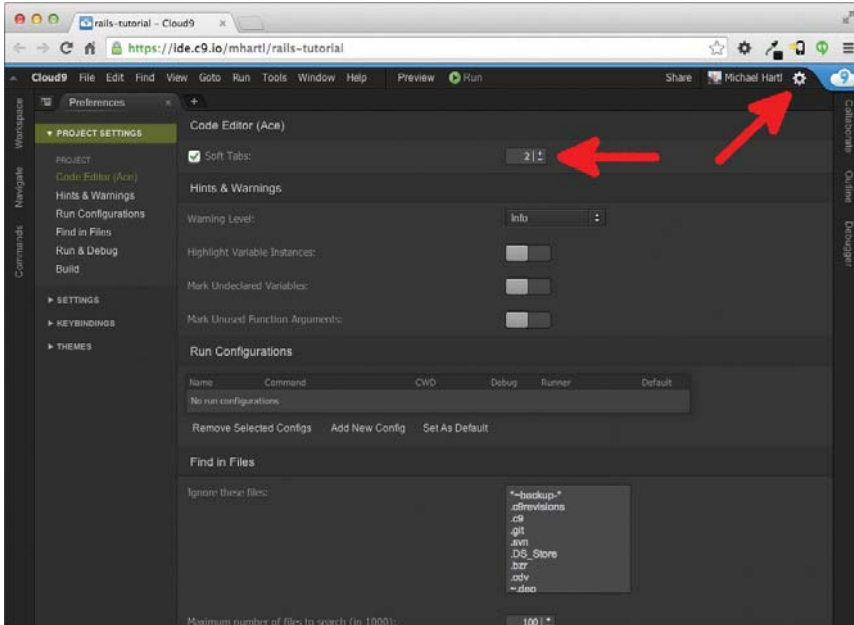


Рис. 1.3 ❖ Настройка использования двух пробелов для отступа в Cloud9

1.2.2. Установка Rails

Среда разработки, представленная в разделе 1.2.1, включает все ПО, необходимое для начала, кроме самого фреймворка Rails¹. Для установки воспользуемся командой `gem` диспетчера пакетов *Ruby Gems*. Введите в командной строке терминала команду, как показано в листинге 1.1. (Если вы работаете в локальной системе, необходимо использовать обычное окно терминала; если в облачной IDE – область командной строки, как на рис. 1.1.)

Листинг 1.1 ❖ Установка Rails с определенным номером версии

```
$ gem install rails -v 4.2.0
```

¹ На данный момент в Cloud9 используется более старая версия Rails, и она несовместима с настоящей книгой, поэтому так важно установить его самостоятельно.

Параметр `-v` гарантирует установку определенной версии Rails. Это очень важно для получения результатов, совместимых с этой книгой.

1.3. Первое приложение

Следуя давним традициям в программировании, нашей первой целью будет создание программы, которая выводит фразу «hello, world» («Привет, мир!»). В частности, мы создадим простое приложение, которое отобразит строку «hello, world!» на веб-странице: в среде разработки на компьютере (раздел 1.3.4) и в Интернете (раздел 1.5).

Практически все приложения Rails начинаются одинаково – с команды `rails new`. Эта удобная команда создает скелет приложения Rails в выбранном каталоге. Тем, кто не стал использовать Cloud9 IDE, рекомендованную в разделе 1.2.1, сначала необходимо создать каталог `workspace` для проекта, если он еще не создан (листинг 1.2), и перейти в него. (В листинге 1.2 показаны команды `cd` и `mkdir`; если вы с ними пока не знакомы, прочитайте блок 1.3.)

Листинг 1.2 ❖ Создание каталога `workspace` для проекта (не нужно в облачной IDE)

```
$ cd                # Перейти в домашний каталог.
$ mkdir workspace  # Создать каталог workspace.
$ cd workspace/    # Перейти в каталог workspace.
```

Блок 1.3 ❖ Краткий курс командной строки Unix

Для читателей, пришедших из Windows или (в меньшей, но все еще значительной степени) из Macintosh OS X, командная строка Unix может выглядеть непривычной. К счастью, если вы используете рекомендованную облачную среду разработки, вы автоматически получаете доступ к командной строке Unix (Linux) в стандартной оболочке интерфейса, известной как Bash¹.

Основная идея командной строки проста: вводя короткие команды, пользователь может выполнить множество операций, таких как создание каталогов (`mkdir`), перемещение и копирование файлов (`mv` и `cp`), навигация в файловой системе за счет смены каталогов (`cd`). Командная строка может показаться примитивной тем, кто знаком в основном с графическим интерфейсом, внешность обманчива: это один из наиболее мощных инструментов в арсенале разработчика. Действительно, крайне редко удается увидеть рабочий стол опытного разработчика, на котором не было бы открыто несколько окон терминала с командной оболочкой.

Вообще говоря, это очень глубокая тема, но для следования за примерами в этой книге понадобится лишь несколько наиболее часто используемых команд, перечисленных в табл. 1.1. Для более глубокого изучения командной строки Unix читайте «Conquering the Command Line»² (Покоряя командную строку) Марка Бейтса (доступны бесплатная онлайн-версия и электронная книга с видеоуроками³).

¹ <https://ru.wikipedia.org/wiki/Bash>.

² <http://conqueringthecommandline.com/book>.

³ <http://conqueringthecommandline.com/#pricing>.