

Моему сыну: Амину Башар Абдул-Джаваду. Я всегда буду любить тебя.

Краткое содержание

Об авторе	xv
О техническом обозревателе	xvii
Благодарности	xix
Введение	xxi

ЧАСТЬ 1 ■■■ Groovy в примерах

■ ГЛАВА 1	Начало работы с Groovy	3
■ ГЛАВА 2	От Java к Groovy	17
■ ГЛАВА 3	Типы данных Groovy и управляющие структуры	45
■ ГЛАВА 4	Объектно-ориентированный Groovy	73
■ ГЛАВА 5	Замыкания	101
■ ГЛАВА 6	Билдеры	117
■ ГЛАВА 7	Работа с базами данных	147
■ ГЛАВА 8	Тестирование с Groovy	163
■ ГЛАВА 9	Разнообразные советы	191

ЧАСТЬ 2 ■■■ Grails в примерах

■ ГЛАВА 10	Начало работы с Grails	213
■ ГЛАВА 11	Web-уровень	227
■ ГЛАВА 12	Уровень данных	263
■ ГЛАВА 13	Скафолдинг	299
■ ГЛАВА 14	Безопасность	329
■ ГЛАВА 15	Тестирование	345
■ ГЛАВА 16	Разнообразные советы	359

Содержание

Об авторе	xv
О техническом обозревателе	xvii
Благодарности	xix
Введение	xxi

ЧАСТЬ 1 ■ ■ ■ Groovy в примерах

■ ГЛАВА 1	Начало работы с Groovy	3
	1-1. Что такое Groovy?	3
	1-2. Что не так с Java?	4
	1-3. Как Groovy устраняет недостатки Java?	5
	1-4. Как загрузить и установить Groovy?	9
	1-5. Какие инструменты поставляются вместе с Groovy?	10
	1-6. Как использовать Groovy-оболочку?	10
	1-7. Как использовать Groovy-консоль?	11
	1-8. Как использовать groovus и groovy?	12
	1-9. Есть ли поддержка Groovy в IDE?	13
	1-10. Как добавить поддержку Groovy в Eclipse?	13
	1-11. Как добавить поддержку Groovy в IntelliJ IDEA?	14
	Резюме	16
■ ГЛАВА 2	От Java к Groovy	17
	2-1. Чем схожи Java и Groovy?	17
	2-2. Чем отличаются Java и Groovy?	19
	Необязательные синтаксические элементы	19
	Новые и усовершенствованные синтаксические элементы, структуры и конструкции	22

Новые хелперы, библиотеки и APIs	29
Другие отличия	33
2-3. Как интегрировать Groovy с Java?	38
Компиляция в байткод	39
Использование GroovyShell	39
Использование GroovyScriptEngine	40
Использование GroovyClassLoader	41
Использование JSR 223	43
Резюме	44

■ ГЛАВА 3 **Типы данных Groovy и управляющие структуры** 45

3-1. Какие виды строк есть в Groovy и как их использовать?	45
3-2. Как использовать регулярные выражения в Groovy?	51
3-3. Чем отличаются числа в Groovy и Java?	54
3-4. Как использовать списки в Groovy?	58
3-5. Как реализовать сортировку слиянием в Groovy?	62
3-6. Как использовать карты в Groovy?	63
3-7. Что такое диапазоны и как их использовать в Groovy?	66
3-8. Что является истинностным значением в Groovy?	67
3-9. Чем отличается оператор switch в Groovy и Java?	68
3-10. Как организовать циклы в Groovy?	69
Резюме	70

■ ГЛАВА 4 **Объектно-ориентированный Groovy** 73

4-1. Чем отличаются классы и скрипты?	73
Один общедоступный класс на файл	74
Несколько классов на файл	74
Только скриптовый код	75
Классы и скриптовый код в одном файле	75
Выбор стратегии	76
4-2. Как использовать пакеты?	76
4-3. Что такое синонимия типов и как её использовать?	77
4-4. Как использовать наследование в Groovy?	78
4-5. Как использовать интерфейсы в Groovy?	79

4-6. Что такое мультиметоды и как их использовать?	82
4-7. Что такое категории и как их использовать?	85
4-8. Чем отличаются поля и локальные переменные в Groovy и Java? ..	85
4-9. Чем отличаются методы в Groovy и Java?	87
Использование позиционных параметров.	88
Использование списка в качестве единственного аргумента	89
Использование массива для необязательных параметров.	89
Использование параметров в виде карты.	90
4-10. Чем отличаются конструкторы в Groovy и Java?	90
4-11. Что такое GroovyBeans?	91
4-12. Что такое GPaths?	93
4-13. Как использовать класс Expando?	95
4-14. Что такое Metaclass и как его использовать?	96
4-15. Как перехватить все вызовы методов объекта?	97
4-16. Как перехватить методы, которые отсутствуют в классе?	98
4-17. Как привнести в класс дополнительное поведение с помощью ExpandoMetaClass?	98
Резюме.	99

■ ГЛАВА 5

Замыкания	101
5-1. Что такое замыкание?	101
5-2. Зачем нужны замыкания?	102
5-3. Можно ли сравнить замыкания с анонимными внутренними классами?	103
5-4. Как создать замыкание?	104
5-5. Как вызвать замыкание?	104
5-6. Как вернуть значение из замыкания?	105
5-7. Как повторно использовать метод в качестве замыкания?	105
5-8. Как передать замыкание в качестве аргумента метода?	106
5-9. Какова область видимости замыкания?	107
5-10. Что означают зарезервированные слова this, owner и delegate внутри замыкания?	109
5-11. Как из замыкания вернуть результат?	110

5-12. Что означает каррирование замыканий?	111
5-13. Как использовать замыкание внутри оператора switch?	113
5-14. Как получить дополнительную информацию о параметрах, переданных в замыкание?	113
5-15. Как использовать замыкания внутри карт?	114
5-16. Как использовать замыкания с файлами?	114
Резюме.	115

■ ГЛАВА 6 **Билдеры** 117

6-1. Что такое билдеры?	117
6-2. Зачем нужны билдеры?	118
6-3. Как использовать MarkupBuilder для генерации XML?	121
6-4. Как использовать MarkupBuilder для генерации HTML?	125
6-5. Как с помощью NodeBuilder создать дерево объектов?	127
6-6. Как использовать ObjectGraphBuilder для генерации дерева объектов?	128
6-7. Как использовать AntBuilder для генерации Ant-заданий?	130
6-8. Как с помощью SwingBuilder создать Swing-виджеты?	131
6-9. Как использовать менеджеры разметки вместе со SwingBuilder? ..	135
6-10. Как добавить экшен к Swing-виджету?	137
6-11. Как совместно использовать экшены из виджетов?	139
6-12. Как использовать Swing-модели?	140
6-13. Как создать свой собственный билдер?	142
Резюме.	145

■ ГЛАВА 7 **Работа с базами данных** 147

7-1. Как установить соединение с базой данных?	147
7-2. Как использовать пул соединений?	149
7-3. Как создать новую таблицу?	149
7-4. Как добавить, обновить и удалить данные?	151
7-5. Как извлечь данные из таблиц?	152
7-6. Как получить метаданные таблицы?	155
7-7. Как использовать DataSet?	156
7-8. Как использовать DataSet с объединением таблиц?	158
Резюме.	161

ГЛАВА 8	Тестирование с Groovy	163
	8-1. Как в Groovy создать встраиваемый тест?	163
	8-2. Как в Groovy создать тестовый класс?	164
	8-3. Как использовать Groovy для тестирования Java-кода?	168
	8-4. Как объединить и выполнить тесты из IDE?	169
	8-5. Как использовать Ant для выполнения тестов?	172
	8-6. Как использовать Maven для выполнения тестов?	173
	8-7. Какие передовые методики тестирования предоставляет Groovy?	175
	8-8. Как использовать карты для тестирования кода?	175
	8-9. Как использовать объект Expando для тестирования кода?	177
	8-10. Как использовать заглушки и макеты в Groovy?	178
	8-11. Как использовать GroovyLogTestCase?	182
	8-12. Как измерить покрытие кода с помощью Cobertura?	184
	Резюме.	189
ГЛАВА 9	Разнообразные советы	191
	9-1. Как использовать шаблоны Groovy для генерации динамического повторно используемого содержимого?	191
	9-2. Как использовать грувелеты для генерации динамического Web-содержимого?	195
	9-3. Как загрузить и обработать XML с помощью XmlParser?	197
	9-4. Как загрузить и обработать XML с помощью XmlSlurper?	201
	9-5. Как использовать XPath?	203
	9-6. Как загрузить XML-содержимое RSS-ленты?	204
	9-7. Как использовать Groovy из командной строки?	205
	9-8. Как использовать ConfigSlurper для создания конфигурационных файлов?	206
	9-9. Как с помощью Groovy запустить внешний процесс?	209
	9-10. Как в Groovy загрузить файл?	209
	9-11. Как обработать все файлы в каталоге?	210
	9-12. Как подсчитать все вхождения слова в строке?	211
	Резюме.	212

ЧАСТЬ 2 ■ ■ ■ Grails в примерах

■ ГЛАВА 10	Начало работы с Grails	215
	10-1. Что такое Grails?	215
	10-2. Зачем нужен ещё один фреймворк?	216
	10-3. Как загрузить и установить Grails?	217
	10-4. Как в Grails создать первое приложение?	218
	10-5. Как использовать Grails в Eclipse?	221
	10-6. Как использовать Grails в IntelliJ IDEA?	222
	10-7. Какие основные команды есть в Grails?	224
	Резюме.	225
■ ГЛАВА 11	Web-уровень	227
	11-1. Как создать контроллер?	228
	11-2. Что такое серверные страницы Groovy?	229
	11-3. Как связаны контроллеры и GSPs?	231
	11-4. Как передать переменные из контроллера в GSP?	232
	11-5. Как использовать теги в качестве вызовов методов?	233
	11-6. Как в контроллере создать несколько экшенов?	234
	11-7. Какие неявные объекты доступны внутри контроллера и GSP? ..	235
	11-8. Как для пользователя создать несколько представлений?	242
	11-9. Как последовательно вызвать экшены?	244
	11-10. Как перехватить экшены контроллера?	245
	11-11. Как выполнить привязку входных параметров?	247
	11-12. Как вернуть в качестве результата JSON?	248
	11-13. Как сохранить доменные классы в виде XML или JSON (сериализовать)?	249
	11-14. Как выгрузить и загрузить файлы?	250
	11-15. Что такое шаблоны и как их использовать?	251
	11-16. Как изменить разметку и внешний вид приложения?	252
	11-17. Как создать собственные пользовательские теги?	257
	11-18. Как использовать фильтры?	258
	11-19. Как использовать Ajax?	260
	Резюме.	262

ГЛАВА 12	Уровень данных	263
	12-1. Как настроить использование базы данных?	263
	12-2. Как создать доменный класс?	267
	12-3. Как установить отношения?	271
	12-4. Как использовать композицию?	278
	12-5. Как выполнять CRUD-операции доменных классов?	279
	12-6. Как выполнять запросы с использованием GORM?	282
	12-7. Как использовать динамические файндеры?	283
	12-8. Как использовать критерии?	285
	12-9. Как использовать HQL?	288
	12-10. Как использовать наследование?	289
	12-11. Что такое оптимистическое и пессимистическое блокирование?	291
	12-12. Как использовать события?	292
	12-13. Как использовать метки времени?	293
	12-14. Как использовать кэширование?	294
	12-15. Как использовать пользовательский идентификатор базы данных?	296
	12-16. Как использовать составной первичный ключ?	296
	12-17. Как добавить индекс к полю?	297
	Резюме.	297
ГЛАВА 13	Скафолдинг	299
	13-1. Как использовать динамический скафолдинг?	300
	13-2. Как динамически выполнить скафолдинг отношений?	303
	13-3. Как настроить созданные представления?	306
	13-4. Какие в Grails есть встроенные ограничения?	310
	13-5. Как переопределить в рамках скафолдинга экшены и представления?	312
	13-6. Как использовать статический скафолдинг?	317
	13-7. Как изменить шаблоны скафолдинга?	321
	13-8. Как добавить собственный редактор свойств?	322
	13-9. Как использовать скафолдинг с классами, отображёнными с помощью Hibernate?	326
	Резюме.	328

ГЛАВА 14	Безопасность	329
	14-1. Как защитить приложение от атак на основе SQL-инъекций?	329
	14-2. Как защитить приложение от межсайтового скриптинга (XSS)? ..	330
	14-3. Как использовать кодеки?	331
	14-4. Как ограничить методы HTTP-запроса, которые могут вызвать экшен?	332
	14-5. Как в приложении реализовать аутентификацию?	333
	14-6. Как использовать плагин AcegiSecurity?	335
	14-7. Как использовать OpenID?	342
	Резюме.	344
ГЛАВА 15	Тестирование	345
	15-1. Как выполнить модульное тестирование приложения?	346
	15-2. Как создать интеграционные тесты?	350
	15-3. Как выполнить тестирование методов render и redirect?	351
	15-4. Как выполнить тестирование библиотек тегов?	353
	15-5. Как выполнить тестирование доменных классов?	354
	15-6. Как создать функциональный тест с помощью Canoo WebTest? ..	356
	Резюме.	358
ГЛАВА 16	Разнообразные советы	359
	16-1. Для чего нужен сервисный уровень?	359
	16-2. Как в Grails можно использовать некоторые передовые возможности Spring?	363
	16-3. Как настроить приложение с помощью внешних файлов?	366
	16-4. Как в приложении настроить журналирование?	369
	16-5. Как использовать Grails вместе с Maven 2?	370
	16-6. Как использовать Grails вместе с REST?	372
	16-7. Как в Grails создать Web-сервисы SOAP с помощью CXF?	375
	Резюме.	382

Об авторе

БАШАР АБДУЛ-ДЖАВАД является ведущим специалистом по программному обеспечению компании Video Monitoring Services (VMS, <http://vmsinfo.com>), которая предоставляет решения в области мониторинга новостей и рекламы. В данной должности Башар перенёс все новые проекты компании с Java и Tapestry на Groovy и Grails. В Нью-Йорке, Аризоне и Ченнае (Индия) Башар обучал разработчиков VMS использованию Groovy и Grails, а также мышлению в терминах Groovy, а не Java. И по сей день Башар всё ещё выступает там на еженедельных обучающих семинарах по темам, относящимся к Groovy, Grails и динамическим языкам.



Получив степень магистра компьютерных наук университета штата Мэн, Башар переехал в солнечный Тусон на работу в Университет штата Аризона в качестве ведущего разработчика Гидрологической информационной системы Аризоны (Arizona Hydrologic Information System, AHIS). AHIS была построена на основе Struts. С растущим разочарованием от излишней сложности Struts и недостатков Java Башар начал поиски более простого и динамического языка, а также фреймворка, который бы функционировал на базе виртуальной машины Java. Вот тогда он и открыл для себя Groovy и Grails.

Когда Башар пришёл в VMS, он принёс с собой и увлечённость Groovy и Grails. В VMS использовали сложный Web-фреймворк: Tapestry. Башар сделал своей целью переход компании на Groovy и Grails, убедив её менеджмент в том, что после нескольких лет нарастающей сложности Tapestry разработчики будут рады работать с Groovy и Grails, да и эффективнее это по крайней мере вдвое.

В дополнение к степени магистра Башар имеет степень бакалавра компьютерных наук Иорданского университета. Также Башар обладает сертификатами Sun-certified Java 1.4 Programmer и Java 1.4 Web Components Developer.

О техническом обозревателе

ДЕЙВ КЛЕЙН является разработчиком компании Contegix, которая специализируется на предоставлении управляемых Интернет-инфраструктур, основанных на Linux, Mac OS X, Java EE и Grails. Дейв вовлечён в разработку корпоративных приложений в течение последних 15 лет. Он работал в качестве разработчика, архитектора, руководителя проекта (не волнуйтесь, он выздоровел), наставника и инструктора. Дейв появляется в различных группах пользователей и на национальных конференциях. Также он является основателем Capital Java User Group в городе Мэдисон штата Висконсин.



Дейв считает себя кочующим программистом. Поработав в Калифорнии, Миннесоте, Техасе и Висконсине, добрался до Миссури. Сейчас он живёт в городе Портидж штата Висконсин со своей женой и 13 будущими консультантами. С размышлениями Дейва по поводу Groovy и Grails можно ознакомиться в его блоге <http://dave-klein.blogspot.com>.

Благодарности

Прежде всего я хотел бы поблагодарить мою семью (маму, папу, брата-близнеца, сестру, её мужа, мою маленькую племянницу и дядю в Чикаго) за их постоянную поддержку, любовь, мудрость, советы, терпение и заботу. Всем, чему я научился в этой жизни, я обязан моим родителям. Без них я бы никогда не был тем, кем являюсь сейчас.

Особенная благодарность моей любимой девушке Лесли за её нескончаемую поддержку во время написания этой книги. При том времени, что уходит на написание книги, я никогда не забуду её понимания и поддержки на протяжении всей работы. Встреча с ней – большая удача в моей жизни.

В Apress я хотел бы поблагодарить Стива Энглина, ведущего редактора по новым поступлениям, за веру в меня и мои способности при написании книги Groovy и Grails. Тома Велша, редактора по развитию, за конструктивную критику моего английского. Дейва Клейна, технического обозревателя, за понимание и советы. Кайли Джонстон, ведущего руководителя проекта, за быстрые напоминания о постоянных задержках материала. Без неё эта книга никогда бы не была опубликована вовремя. Также хотел бы поблагодарить Шарон Вилки (корректор) и Келли Гюнтер (выпускающий редактор). Все, с кем я работал в Apress, были очень дружелюбны, деликатны и преданы своей работе.

Также я хотел бы поблагодарить моих коллег из VMS. Скотта Сигала, моего менеджера, за его одобрение Groovy и Grails и предоставление мне возможности использовать их в работе. Джерри Лью, директор по информационным технологиям, за то, что прислушался к рекомендациям Скотта по Groovy и Grails. И Криса Тилмана за чтение корректуры первых трёх глав этой книги и понимание.

Наконец, я должен поблагодарить талантливых в Groovy и Grails людей: Дирка Кёнига, Эндрю Гловера, Пола Кинга, Гийома Лафоре и Джона Скита – авторов *Groovy in Action*, очень полезного руководства при написании этой книги, и Грэма Роше – основателя Grails и автора *The Definitive Guide to Grails*. Спасибо большое за столь изумительный Web-фреймворк. Также я хотел бы поблагодарить очень активное сообщество Groovy и Grails. Ваша помощь в виде почтовых рассылок оценена по достоинству.

Введение

Платформа Java топчется на месте довольно долгое время. При этом язык программирования Java начинает устаревать. Сейчас для Java-разработчиков наступает время к переходу в сторону мышления в терминах динамических языков. Groovy является одним из самых лучших динамических языков доступных на платформе Java. После нескольких лет работы с Groovy я твёрдо убедился, что всем Java-разработчикам следует по крайней мере познакомиться с Groovy. Количество кода, которое можно сократить с помощью динамического языка подобного Groovy, действительно поражает, особенно, работая с коллекциями и файлами. Именно по этой причине я решил написать эту книгу. Я хочу поделиться с Java-разработчиками огромным выигрышем в эффективности, которая достигается с помощью Groovy.

Динамические языки подобные Groovy делают реальностью такие фреймворки как Grails. Grails – это глоток свежего воздуха для Java-разработчиков и это одна из основных причин моей заинтересованности динамическими языками. Я помню первые дни разработки на Java с использованием Struts и Tapestry. Но я по ним не скучаю. Лично мне эти фреймворки всегда казались излишне сложными. Я просто не мог определить объёмы конфигурационного и шаблонного кода, необходимые для того, чтобы хоть что-то сделать. Это не те фреймворки, о которых стоит думать. Фреймворки должны делать вещи проще и позволять сосредоточиться на логике. Это то, что делает Grails. Grails является осмысленной технологией, что для меня является первым, что я ищу в новой технологии. Grails является столь простым и одновременно мощным фреймворком, что нельзя не удивляться, почему о нём никто не подумал раньше.

Одна из сильных сторон Groovy и Grails – это то, что виртуальная машина Java является для них «родной». Учитывая вездесущность Java в наше время, было бы глупо предлагать Java-разработчикам отказаться от всей Java-инфраструктуры, APIs, библиотек и фреймворков и начать всё сначала. По этой причине Groovy и Grails обязаны быть успешными в мире корпоративных приложений, где платформа Java успешно обосновалась. Их идеальная интеграция с Java – огромное подспорье в распространении. В одной из организаций мы обсуждали, следует ли использовать Ruby on Rails или Groovy и Grails. К концу дня победу одержали Groovy и Grails. Их отличное взаимодействие с Java и лёгкость изучения для Java-разработчиков стали ключевыми факторами, повлиявшими на окончательное решение.

В этой книге у меня две цели. Во-первых, на практике научить Groovy и Grails с нуля. Во-вторых, предоставить практические решения частых проблем в Groovy и Grails. Я хотел бы, чтобы Вы могли взять книгу и, выбрав интересующую проблему, быстро найти

подходящее решение. Вы никогда не найдёте подробных теоретических выкладок о том, как же всё устроено изнутри, а непосредственно небольшие фрагменты кода для решения проблемы.

Я надеюсь, Вы получите такое же удовольствие от прочтения книги, как я при её написании. С Groovy и Grails действительно удобно работать. Я не припомню, чтобы получал от работы с технологией такое удовольствие, как от работы с Groovy и Grails.

Для кого эта книга

Эта книга нацелена напрямую Java-разработчикам. Она не подразумевает предварительного знакомства с Groovy и Grails. И будет преподносить материал в виде вопросов и ответов. У разработчиков, не использовавших Java, но знакомых с динамическими языками (такими как Ruby, PHP или Python), также не должно возникнуть проблем при чтении книги, хотя Java-разработчики, вероятно, извлекут из неё больше пользы.

Grails – это больше, чем Web-фреймворк. Это множество приложений, которые собирают вместе другие технологии, а именно: Hibernate, Spring и SiteMesh. Хотя по этим технологиям и не требуется предварительных знаний, читателям, которые уже знакомы с ними, будет легче понять соответствующие советы, чем тем, которые их никогда не использовали.

Как организована эта книга

Эта книга разделена на 16 глав с помощью подхода в виде вопросов и ответов.

Я всегда был поклонник книг в виде советов. Они раскрывают суть без напрасной траты времени читателя. Именно это делает данная книга. Она состоит из двух частей: Groovy и Grails. Часть, посвящённая Groovy, включает первые девять глав.

Глава 1 представляет краткое введение в Groovy, показывает варианты его использования и по шагам объясняет, как загрузить и установить Groovy на компьютере.

Глава 2 в основном предназначена для того, чтобы облегчить Java-разработчикам переход от синтаксиса Java к Groovy. Показаны главные сходства и отличия.

Глава 3 обсуждает типы данных и управляющие структуры. Типы данных Groovy включают простые типы данных и коллекции. Управляющие структуры состоят из циклических структур и условных структур.

Глава 4 показывает Groovy с объектно-ориентированной стороны. Groovy – это полностью объектно-ориентированный язык, поэтому Java-разработчики будут чувствовать себя как дома.

Глава 5 касается темы, которая, возможно, наиболее трудна для понимания Java-разработчиками: замыканий. На примерах эта глава пытается раскрыть загадку замыканий и показать варианты их использования.

Конец ознакомительного фрагмента.

Приобрести книгу можно
в интернет-магазине «Электронный универс»
(e-Univers.ru)