



From Technologies to Solutions

# Apache OFBiz Development

## The Beginner's Tutorial

Using Services, Entities, and Widgets to build custom ERP and CRM systems

Jonathon Wong

Rupert Howell

**PACKT**  
PUBLISHING

# Table of Content

<b>Preface</b>	<b>1</b>
<b>Chapter 1: Getting Started with OFBiz</b>	<b>9</b>
<b>Getting the OFBiz Code</b>	<b>9</b>
<b>Downloading and Installing SVN</b>	<b>10</b>
Downloading TortoiseSVN	10
Using SVN to Get OFBiz	11
Our OFBiz Workspace—First Look	13
<b>Installing the Java Development Kit (JDK)</b>	<b>14</b>
Downloading JDK 5.0	14
Installing JDK 5.0	15
<b>Downloading OFBiz Ready to Launch</b>	<b>16</b>
<b>Setting Up an Eclipse Project</b>	<b>16</b>
<b>Using Derby—the OFBiz Stock Database</b>	<b>19</b>
Compiling OFBiz and Loading the Data	20
Verifying the Installation Process	20
The Compilation Portion	20
The Data Loading Portion	21
Some Pre-Ignition Processes we won't do Twice	21
Backing Up the Derby Data Files	22
<b>Running OFBiz</b>	<b>22</b>
Allocating Memory for OFBiz	22
Starting OFBiz and Verifying the Start-Up Process	24
Seeing the Logs in Real Time	25
Possible Problems with Start-Up	25
<b>Switching Off OFBiz</b>	<b>25</b>
<b>Our First Tour of OFBiz</b>	<b>26</b>
Accessing OFBiz	26
Exploring the Webapp "ecommerce"	28
Let's Buy Something	28
Exploring the Webapp "order"	32

Receiving Payment	32
Fulfilling the Order	34
Invoice Automatically Generated with Payment Applied	34
End-to-End Tour Completed	34
<b>Summary</b>	<b>35</b>
<b>Chapter 2: Working with OFBiz</b>	<b>37</b>
<b>Adding Our First Field</b>	<b>37</b>
Changing the Data	39
Editing the Entity Definition	39
Updating the Database	40
Changing the Looks	40
Editing the User-Interface	41
Checking Our Changes	41
Changing the Flow	42
Rewiring the "Save" (Update Postal Address) Button	42
Creating the New Widget Screen	43
Creating the FTL File	44
More to the Flow	45
Some Changes Possible with Engines Running	47
<b>Resetting Our Play Area Quickly</b>	<b>48</b>
Skipping Some Pre-Ignition Processes	49
Restoring Derby Data Files	49
Removing the Web Server (Catalina) Work Files	49
Updating the Database with Our Data Entity Changes	49
Showing Off Our Spanking New OFBiz Installation	50
Tripping Up Our Plan	50
Storing a Save-Point to Dramatically Ease Testing	50
Archiving Derby Data Files and Web Server Work Files	51
Restoring a Save-Point	52
Restoring the Derby Data Files	52
Restoring the Web Server Work Files	52
Computer, Run Scenario A from Last Save-Point	52
<b>The Structure of OFBiz in General</b>	<b>53</b>
Components in OFBiz	53
Referencing Components in OFBiz	54
Creating Our Own OFBiz Component	55
Creating the Component	55
Using Our Component	55
Cleaning Up Our Mess in the "party" Component	56
Converting the BeanShell to a Java Event	57
Clean Extension Strategies Employed	59
Checking that Our Move was Successful	60
A Bit More Mess Remains	60
Webapps in OFBiz	60
Creating Our First Webapp	61

---

Webapp URIs in this Book	62
Testing Our First Webapp	63
<b>The Model-View-Controller Architectural Pattern</b>	<b>63</b>
The MVC in Plain English	63
The Model in OFBiz	64
The View in OFBiz	64
The Controller in OFBiz	65
Other Files in an OFBiz Component	66
<b>Summary</b>	<b>67</b>
<b>Chapter 3: Screen Widgets</b>	<b>69</b>
<b>Equipping Our Webapp with a Screen Widget View Handler</b>	<b>69</b>
<b>Using the Screen Widget View Handler</b>	<b>71</b>
<b>Files and Locations</b>	<b>71</b>
<b>Creating Our First Screen Widget</b>	<b>71</b>
Defining a Screen Widget	71
Informing the Control Servlet about the Screen Widget	72
Referencing Screen Widgets	73
Uniform Pattern of Flow in OFBiz	73
Seeing Our First Screen Widget	74
<b>The Anatomy of the &lt;section&gt; Element</b>	<b>74</b>
Our First Conditional Screen Widget	74
Element Order in the controller.xml	75
Inform the Control Servlet about the Screen Widget	75
If-Then-Else Structure	75
The <if-condition> Element	76
The Then—<actions> and <widgets> Elements	76
The Else—<fail-widgets> Element	78
The Minimum <section>	78
Sending Parameters with Requests	79
Seeing Our Conditional Screen Widget	79
Screen Widget Context and Variables	79
Utility Objects in Context	80
<b>Nested Sections for Nested Conditions</b>	<b>85</b>
<b>Organizing a Large Screen into Smaller Screens</b>	<b>88</b>
The Global Context Revisited	90
Outer Contexts Visible to Nested Ones	92
<b>Screen Widget's Integration with FreeMarker</b>	<b>93</b>
<b>Cleaning Up in the "party" Component</b>	<b>94</b>
<b>Commenting Changes to the Core Code</b>	<b>96</b>
<b>Screen Widgets as Templates</b>	<b>97</b>
A Candidate for Templating	97
Creating the Header	98
Creating the Footer	98

Using Our Header and Footer	99
Seeing Our First Well-Formed XHTML Document	99
Using Decorator Screen Widgets for Templating	100
Creating a XHTML Decorator Screen	100
Using the XHTML Decorator Screen	100
Seeing Our First Decorator in Action	101
Multiple Content Slots	102
Creating the First Half of the Header	102
Creating the Second Half of the Header	102
Adding a Content Slot to a Decorator	103
Using Our Multi-Slot Decorator	103
Seeing Our First Multi-Slot Decorator	104
Nesting Decorators	104
Top-Down Approach (delegation)	104
The Bottom-Up Approach (Vertical Stack)	106
Using Both Approaches	107
<b>Summary</b>	<b>109</b>
<b>Chapter 4: Form Widgets</b>	<b>111</b>
<b>Files and Locations</b>	<b>112</b>
<b>Creating Our First Form Widget</b>	<b>112</b>
Creating the Containing Screen Widget	112
Referencing Form Widgets	113
Create the Form Widget	113
Seeing Our First Form	114
Understanding the Form Attributes	114
Minimum Requirements to Use Form Widgets	115
Including the Minimal Requirements	116
<b>Form Processing via Request Event</b>	<b>116</b>
<b>Java Events</b>	<b>117</b>
<b>Submitting and Processing Our First Form</b>	<b>118</b>
<b>The "list" Type Form Widget</b>	<b>119</b>
Creating the Containing Screen	119
Adding Form Processing Code	120
Publishing the Form	121
Seeing Our First "list" Type Form	122
<b>The "multi" Type Form Widget</b>	<b>122</b>
Creating the Containing Screen	122
Loading Data for the Form	123
Publishing the Form	124
Creating the Form-Processing Logic	125
Seeing Our First "multi" Type Form	125
<b>Alternative Targets in Two-Target Forms</b>	<b>126</b>
Creating the Form	126

---

Creating the Containing Screen	127
Publishing the Two-Target Form	127
Seeing Our First Two-Target Form	128
<b>Row-Level Actions</b>	<b>128</b>
Creating the Form	129
Creating the Containing Screen	129
Publishing the Form	130
Seeing the Form in Action	130
<b>Summary</b>	<b>131</b>
<b>Chapter 5: Other View Element Types in Screen Widgets</b>	<b>133</b>
<b>Menu Widgets</b>	<b>134</b>
Creating Our First Menu Widget	134
Including Our Menu Widget in Our Screen Widgets	135
Understanding the Menu Item Attributes	135
Including the Menu Widget via a Decorator Screen Widget	135
Add Screen Widget "ConditionalScreen" to the Menu	136
Sub-Menus and Conditional-Menu Items	137
Pre-processing Actions for Menu Widgets	139
<b>FreeMarker</b>	<b>142</b>
As Decorator Templates	142
Displaying Dynamically Created List Variables	143
Iterating Through the List	144
<b>Bringing it All Together</b>	<b>145</b>
The User-Interface Labels	146
Adding the appheader	148
<b>Summary</b>	<b>150</b>
<b>Chapter 6: The Controller</b>	<b>151</b>
<b>How OFBiz Hears Our Requests—The Control Servlet</b>	<b>151</b>
Defining a Control Servlet for a Webapp	152
Using the Control Servlet	153
Funnelling All Requests to the Single Control Servlet	153
Defining Needed Utility Objects for the Control Servlet	154
GenericDelegator Object	154
The GenericDispatcher Object	155
Some Background on Servlets	156
<b>Programming a Control Servlet</b>	<b>156</b>
Logging into Our Learning Application	156
Specifying Handlers	159
Request Maps	160
Knocking on the Right Doors	160
Security—Before Answering the Door	160
The https Attribute	160

The auth Attribute	162
The direct-request Attribute	163
Event—Determining a Response	165
Java Events	165
Response—Defining Various Responses	166
View Maps	172
<b>Summary</b>	<b>173</b>
<b>Chapter 7: Entities, View Entities, and Extended Entities</b>	<b>175</b>
<b>Entities</b>	<b>176</b>
The Structure of the Data Model	176
Referencing Fields of an Entity	176
OFBiz Uses Relational Database Management Systems	176
Curious Trivia about the Relational Model	177
Entity Engine Concepts	177
Datasources	177
Entity Delegates	178
Entity Groups	179
Defining Our First Entity	179
Assigning Our Entity to an Entity Group	180
Loading Our Entity into the Entity Engine	180
Seeing Our First Entity	181
Using Our First Entity	183
Creating a Drop-Down	184
Populating the Drop-Down	186
Expiring a Value	187
Un-Expiring a Value	187
Anatomy of an <entity> Element	187
The <field> Element	188
The <prim-key> Element	189
The <relation> Element	189
The <index> Element	194
Relation Types	196
One-to-One Relations	196
One-to-Many Relations	196
One-to-One Relations with No Foreign Keys	199
<b>View Entities</b>	<b>200</b>
Anatomy of a <view-entity>	201
The <member-entity> Element	201
The <alias> and <alias-all> Elements	202
The <view-link> Element	206
The <relation> Element	209
Applying Functions on Fields	209
Counting the Number of Records	209
Counting Distinct Records	210

---

Arithmetic Aggregate Functions	210
Uppercase and Lowercase Functions	210
Grouping for Summary Views	211
Complex Aliases	211
Nested <complex-alias> Elements	212
<b>Extending Entities</b>	<b>213</b>
<b>Summary</b>	<b>214</b>
<b>Chapter 8: Accessing the Entities and View Entities</b>	<b>217</b>
<b>Setting-Up Our Playground</b>	<b>218</b>
The Script Processor	218
The Generic Screen	219
Creating the Screen Widget	219
Creating the Form Widget	220
Creating the FreeMarker File	220
Publishing Our Generic Screen	222
Testing Our Playground	222
<b>GenericValue Objects</b>	<b>223</b>
<b>Creating a Database Record</b>	<b>224</b>
Why a Map is Used?	224
<b>Updating Database Records</b>	<b>225</b>
<b>Deleting Database Records</b>	<b>226</b>
<b>Retrieving Database Records</b>	<b>226</b>
Find Records by Conditions	227
Conditions	228
Comparison Operators (EntityComparisonOperator)	228
Condition Lists	231
Condition Joiners (EntityJoinOperator)	232
Tools for Common Styles of Conditions	238
Conditions Joined by AND	238
Conditions Joined by OR	239
Counting the Number of Records Retrieved	240
OFBiz Date Condition	241
Getting Related Records	243
One-to-One Relations	243
One-to-Many Relations	244
Utilities for Post-Query processing	245
Post-Query Ordering	246
Post-Query Filtering by Conditions	247
Post-Query Filtering by Date	248
Other Post-Query Filtering Methods	250
<b>Using the Entity Engine Cache</b>	<b>251</b>
<b>Dynamic View Entities</b>	<b>252</b>
Left Outer Joins with the Dynamic View Entity	256
Performing the Lookup	257



EntityFindOptions	257
<b>Paginating Using the EntityListIterator</b>	<b>258</b>
Paginating through Party Records: A Working Example	258
<b>Functions and Dynamic View Entities</b>	<b>263</b>
<b>Summary</b>	<b>263</b>
<b>Chapter 9: The Events</b>	<b>265</b>
<b>Java Events</b>	<b>265</b>
Security and Access Control	266
User Logins are like Access Cards	267
Security Groups are like Access Levels	269
Security Permissions are like Individual Secured Areas	270
Security Permissions are Contained within Security Groups	270
User Logins are Assigned Security Groups	271
Dealing with Security in Java	271
Sending Feedback to the End-User	274
Conventions for Message Placeholders	274
Testing the Conventions	275
Handling Parameters	277
Accessing Localized Messages	279
Parameterizing Messages	281
Catering for Multiple Languages	282
<b>Working with the Database</b>	<b>284</b>
<b>Summary</b>	<b>284</b>
<b>Chapter 10: The Service Engine</b>	<b>287</b>
<b>Defining a Service</b>	<b>288</b>
<b>Creating the Java Code for the Service</b>	<b>288</b>
<b>Testing Our First Service</b>	<b>289</b>
<b>Service Parameters</b>	<b>291</b>
Input Parameters (IN)	291
Output Parameters (OUT)	293
Two Way Parameters (INOUT)	294
Special Unchecked Parameters	295
Optional and Compulsory Parameters	295
<b>The DispatchContext</b>	<b>296</b>
<b>Service Security and Access Control</b>	<b>297</b>
<b>Calling Services from Java Code</b>	<b>299</b>
<b>Implementing Interfaces</b>	<b>303</b>
Overriding Implemented Attributes	304
<b>Synchronous and Asynchronous Services.</b>	<b>304</b>
<b>Using the Job Scheduler</b>	<b>305</b>
<b>Quickly Running a Service</b>	<b>307</b>
<b>Naming a Service and the Service Reference</b>	<b>307</b>

---

<b>Event Condition Actions (ECA)</b>	<b>308</b>
Service Event Condition Actions (SECAs)	308
Entity Event Condition Actions (EECAs)	310
<b>Summary</b>	<b>311</b>
<b>Chapter 11: Permissions and the Service Engine</b>	<b>313</b>
<b>Simple Permissions</b>	<b>313</b>
<b>Two-Part Permissions and Special "_ADMIN" Permissions</b>	<b>316</b>
<b>Role Checks</b>	<b>317</b>
<b>Combining Multiple Checks</b>	<b>319</b>
<b>Nested Checks</b>	<b>320</b>
<b>Complex Permissions</b>	<b>321</b>
Setting-Up Our Playground	321
Creating the Request and View Maps	322
Creating the Screen and Form Widgets	322
Creating the Entity <b>PlanetReview</b>	<b>324</b>
Defining Services that Require Complex Permission	325
Implementing Services that Require Complex Permission	325
Defining Permission Services	327
Implementing Permission Services	328
Playing with Complex Permissions	331
Complex Permissions and Simple Permissions cannot be combined	335
<b>Summary</b>	<b>336</b>
<b>Chapter 12: Minilang</b>	<b>337</b>
<b>What is Minilang?</b>	<b>338</b>
Tools to Code XML	339
<b>Defining a Simple Service</b>	<b>339</b>
Defining the Simple Service	340
Writing the Simple Method	340
<b>Simple Events</b>	<b>342</b>
<b>Validating and Converting Fields</b>	<b>343</b>
Validating a Simple Event Example	344
<b>Checking Security in Minilang</b>	<b>348</b>
<b>Invoking from Minilang</b>	<b>349</b>
Calling Services from Minilang	349
Calling Simple Methods	349
Calling Java Methods	350
Calling BeanShell	350
<b>Minilang in Screen Widgets</b>	<b>350</b>
<b>Summary</b>	<b>351</b>

<b>Chapter 13: Tying Up the Loose Ends</b>	<b>353</b>
<b>The OFBiz Look and Feel</b>	<b>353</b>
Changing the Customer Facing Site	355
Changing the Back Office Screens	357
The Header	358
The Applications Bar (appbar)	358
The Central Area	359
The Footer	360
<b>Using FreeMarker</b>	<b>360</b>
OFBiz Transform Tags	362
<@ofbizUrl>	362
<@ofbizContentUrl>	363
<@ofbizCurrency>	364
Calling Java from FreeMarker	364
<b>OFBiz Utilities</b>	<b>365</b>
UtilMisc	366
UtilValidate	366
UtilDateTime	366
Debug	366
<b>Outputting Different Formats</b>	<b>367</b>
Outputting a PDF	367
<b>Summary</b>	<b>370</b>
<b>Chapter 14: Tips and Techniques</b>	<b>373</b>
<b>Debugging Techniques</b>	<b>373</b>
Logging	374
console.log	374
ofbiz.log	374
Configuring the Logs	374
Viewing the Logs through Webtools	375
Writing Information to the Logs	375
Logging Minilang	376
<b>Debugging Java Code</b>	<b>378</b>
Passing in the VM Parameters	378
Configuring the Remote Debugger	379
<b>Managing a Project Using Subversion</b>	<b>382</b>
Preparing the Repository and Creating the Project	383
Step 1: Checking Out the Latest OFBiz Code	384
Step 2: Making a Copy	384
Step 3: Importing Our Copy	385
Step 4: Branching ofbiz\current to Create the Trunk	387
Getting the Latest Features and Bug Fixes	388
Step 5: Updating OFBiz Current	388
Step 6: Merging Changes between Revisions of OFBiz Current into Our Project	391
<b>Apache HTTP Server and OFBiz using mod_proxy_ajp</b>	<b>392</b>

---

Installing the Apache HTTP Server	392
What is mod_proxy_ajp?	393
Configuring Apache HTTP Server to Use mod_proxy_ajp	393
Configuring OFBiz to Use the AJP Connector	394
Getting the Secure Pages to Work	395
Configuring SSL in Apache	395
Creating a Self-Signed SSL Certificate	396
Going into Production	396
<b>Development Tips</b>	<b>397</b>
<b>Summary</b>	<b>398</b>
<b>Appendix A: Simple Method User's Guide</b>	<b>401</b>
<simple-methods>	401
<simple-method>	401
Call Operations	402
<call-map-processor>	402
<call-service> / <call-service-async>	403
<set-service-fields>	404
Handling the Results of Service Calls	404
<results-to-map>	404
<result-to-field>	405
<result-to-request>	405
<result-to-result>	406
<call-bsh>	407
<call-simple-method>	407
<call-object-method>	408
<call-class-method>	409
Service Operations	409
<field-to-result>	410
Event Operations	410
<session-to-field>	410
Environment Operations	411
<set>	411
<clear-field>	412
<first-from-list>	412
Miscellaneous Entity Operations	412
<sequenced-id-to-env>	413
<make-next-seq-id>	413
Entity Find Operations	414
<entity-one>	414
<entity-and>	415
<entity-condition>	416
Sub-Elements	416
<condition-list>	416
<having-condition-list>	417
<select-fields>	417
<order-by>	417

<entity-condition> complete example:	417
<entity-count>	418
<get-related-one>	418
<get-related>	419
<order-value-list>	420
<filter-list-by-and>	421
<filter-list-by-date>	421
<b>Entity Value Operations</b>	<b>422</b>
<make-value>	422
<clone-value>	423
<create-value>	423
<store-value>	424
<refresh-value>	424
<remove-value>	424
<remove-related>	425
<remove-by-and>	426
<set-pk-fields> / <set-nonpk-fields>	426
<store-list>	427
<b>Control Operations</b>	<b>427</b>
<iterate>	427
<iterate-map>	428
<check-errors>	428
<add-error>	429
<return>	429
<b>If Conditions</b>	<b>430</b>
<if-validate-method>	430
<if-instance-of>	430
<if-compare>	431
<if-compare-field>	432
<b>Conditions:</b>	<b>433</b>
<b>Other Operations</b>	<b>434</b>
<log>	434
<now-timestamp-to-env>	434
<now-date-to-env>	434
<set-current-user-login>	434
<calculate>	435
<b>Index</b>	<b>437</b>

---

# Preface

Apache Open For Business or OFBiz as it is more commonly known, is an open source framework designed to facilitate the building of Enterprise Resource Planning (ERP) software. ERP is a general name for any system which attempts to integrate all business processes and underlying data into one single system. Indeed the OFBiz framework not only facilitates the building of your own custom software, but also comes packaged with many tools you would expect from an ERP system, and much more. The extent to which you wish to use these applications is entirely up to you and the needs of your business. Some businesses choose to use some or all of these components virtually straight out of the box. Others may spend time and money customizing these components or building new ones to suit their own needs and their own unique business processes. Since OFBiz is licensed under the Apache License Version 2.0, organizations can use, customize, extend, modify, repackage, and even resell OFBiz completely free of charge.

OFBiz is aimed primarily at ecommerce businesses, giving easily customizable tools such as a full Warehouse Management System (WMS), an accounting system and full order and product management systems. It even has a full front end, customer facing website and shopping cart with tools and features comparable to multimillion dollar websites such as Amazon, not to mention its own set of self maintenance and administrative tools. Out of the box, OFBiz is a multi-currency system working just as well with British Pounds, Euros, or any other currency as it does with US Dollars. It is multilingual and is able to display text in different languages depending on where in the world the user or customer is looking. It is so versatile it is not even tied to one database, and fully supports most well known databases.

The main reason for its versatility and size has been its open source model. OFBiz is truly a collaborative effort with a small number of committers who have volunteered to develop and maintain a code base supplied by both themselves and a growing community. Although documentation on the tools is often thin on the ground (this is mainly because of the speed at which the project and components evolve), there are free and active mailing lists set up that will become an invaluable learning tool and source of information as you progress with OFBiz. The OFBiz project employs the use of the well known JIRA application (a bug and issue tracking and project management tool - which is using the OFBiz Entity Engine, a major part of the framework). This allows developers and users to tell the community about any bugs they find in the software or request new features that they would find handy but perhaps don't have the resources to develop for themselves. Who knows? Once you have read this book you may even want to have a go at developing an outstanding issue or fixing a bug for the project yourself!

## **A Brief History of OFBiz**

OFBiz was created in May 2001 by David E Jones and Andy Zeneski after they discovered that the problems they had tried to solve independently were similar and that existing solutions at the time were not sufficient. They realized that these problems could be solved using an Open Source model.

During the course of the next 5 years the OFBiz community grew and as such the number of organizations adopting OFBiz as their ERP system rose, and in January 2006 the project was accepted by the Apache Incubator Project Management Committee (PMC).

In December 2006 the Board of Directors voted to make Apache OFBiz a top-level project. Today OFBiz remains a highly active, community driven, not for profit project enjoying a growing membership and a wider adoption.

## **About This Book**

The examples and tutorials featured within this book aim to, by the end of our journey, come together to produce a working web-based application. Various approaches will be explored during this project. Different combinations of techniques will be tried and evaluated, so we can learn when best to use them.

We will find that OFBiz is more than just a framework or toolkit. A good portion of OFBiz is comprised of ready-made functions and structures that easily lend themselves to being building blocks for ERP software, the application components. Whilst it is possible to use only the OFBiz framework to build a complete ERP application, it will be at least several magnitudes faster to build on top of these ready-made application components. We will study some of these applications while doing our project, which will also lead to a better understanding of some tried and tested best practices.

## What This Book Covers

*Chapter 1*, using a Windows machine, guides us through downloading and installing the necessary software we need to obtain and run the OFBiz framework. We are shown how to create an Eclipse project and once running we are shown a few of the components as we place an order on the applications customer facing website and fulfill the order using the back office's Order Manager component.

In *Chapter 2* we learn the structure of OFBiz. We are introduced to the concepts of the framework, applications, and hot-deploy directories. We perform our first customization on an existing OFBiz component and finally create the structure of our own bespoke application.

In *Chapter 3* we take a look at how the output to the screen is constructed using the screen widgets. We start by creating a simple screen in our learning component, showing a basic output. By the end of the chapter we have learned how to create complex screens, made up of different sections.

In *Chapter 4* we study form widgets. We learn how they are used within screen widgets and can save us development time and effort by quickly producing XHTML forms so we can input information to the application.

In *Chapter 5* we complete our investigation into the presentation layer of OFBiz by learning how to use Menu-Widgets to navigate around our component. We also take more of a look at how FreeMarker can help us display more complicated screens.

In *Chapter 6* we re-visit the Control, learning more about how OFBiz makes use of the Front Controller pattern to configure the flow through our component in just one place. We learn how OFBiz handles different types of requests and we are introduced to the concept of security. By the end of the chapter, we have added "log in" functionality to our bespoke application and have seen how easy it is to force a request to be "secure".



In *Chapter 7* we move on to the concepts of the Entity Engine and learn how OFBiz employs the use of the Delegation Pattern to give us easy access to methods to persist data. We learn how OFBiz creates the database structure, adding fields, tables, constraints and indexes from definitions in XML files. We see how, by using View Entities, we can perform joins across tables allowing us to create complex queries. We are also introduced to the Webtools administration component of OFBiz and discover how to access the raw data through these screens.

In *Chapter 8* we are led through a series of examples designed to showcase data lookup and persistence techniques. We learn how to use the GenericDelegator's methods to lookup and manipulate the underlying data. We discover how using the Entity Engine Cache can massively improve performance by cutting down the number of database queries and learn how complex queries can be created on the fly by using Dynamic View Entity. Finally we learn how to use the EntityListIterator to efficiently paginate through large record sets.

In *Chapter 9* we take a closer look at Java events by learning a number of techniques vital to programming the flow of the application. We also take a look at how we can assign users permissions and how these permissions are checked within the Java methods.

In *Chapter 10* we next see another type of event and a very important one – the services. We learn about the advantages of the Service Engine and how it works, learning how to define and write services in Java. We learn the difference between invoking these services synchronously and asynchronously and how services can be scheduled using the Job Scheduler. Finally we learn how to trigger these services using ECAs (Event Condition Actions).

In *Chapter 11* we move on to study complex permissions, learning how to assign users granular permissions, and how simply these permissions can be checked from our services. We learn by example how to restrict users from viewing or inputting data depending on access rights whilst building up our bespoke application.

In *Chapter 12* we learn about the OFBiz Mini-Language. We see how we can write simple services and events in Minilang, and we learn when it should be ideally used. We see its versatility and see how widespread its concepts are used throughout the framework.

In *Chapter 13* we come towards the end of learning about the framework, we see how easy it is to change the look and feel of the component and study the structure of the existing screens. The chapter moves on to some more advanced FreeMarker techniques that are commonly used throughout all of the components. Finally using the production of a PDF as an example we see how to output different formats.

In *Chapter 14* we learn some real world developing techniques, including how to debug through the different parts and languages found within the framework. We see how to connect a remote debugger to the application and step through the Java code line by line using the Eclipse IDE. We next learn the concepts behind getting the latest bug fixes and features and merging these into our project using Windows tools enabling us to successfully work with the latest and greatest version on OFBiz. Finally we see learn how to run OFBiz behind the Apache HTTP Server allowing us to create a scalable architecture.

## What You Need for This Book

Full download and installation instructions for all software and applications required to obtain and run OFBiz can be found in Chapter 1. These included:

- The Java Development Kit
- TortoiseSVN
- Eclipse IDE

At the time of writing, the latest stable version of OFBiz is 4.0. We will be working with that version, and not with the trunk version. In OFBiz speak (version control speak, rather), such a stable version is termed a "release branch"; the single trunk that forges ahead 24/7 is simply called the "trunk". Each stable branch can still move forward, due to the application of bug fixes. Hence, even release branches do go through revisions.

The trunk is the "latest, greatest and riskiest" state of OFBiz, where new features are added frequently, possibly along with new bugs. Such frequent addition of features do not allow for adequate testing. Before the dust can settle after a new addition by one contributor, more additions may come in from other contributors around the world. Scary as that sounds, this rapid and globally collaborative environment gives the OFBiz framework the fantastic advantage of being improved at a breakneck pace.

On the other hand, release branches like OFBiz 4.0 have had a "feature-freeze", and are invested with testing efforts focused on removing bugs and enhancing stability rather than adding new and untested features. Release branches are ideal for deployment in production environments (for real business use) where stability and ease of maintenance and support are vital.

Working with the trunk presents a moving target, a challenge that is best taken when we have some decent competency with the OFBiz framework as well as familiarity with recent OFBiz trends and changes.

This book may refer to existing code and also to line numbers within files. Bug fixes may therefore have an effect on these line numbers. Since they are only bug fixes, the line numbers should not drastically alter and a simple "Find" in the file will point you in the right direction.

The official policy on the frequency of releases (creation of new release branches) is published once per year, as defined in the Apache OFBiz Release Plan - General Release Policies <http://docs.ofbiz.org/display/OFBADMIN/Release+Plan#ReleasePlan-GeneralReleasePolicies>.

## Who is This Book For

This book is intended for people wishing to understand and begin to customize the OFBiz framework. A basic level of Java is required and some understanding of Object Oriented Design (OOD) concepts would be beneficial. A basic grasp of Simple Query Language (SQL), although not strictly necessary, is certainly helpful to quickly understand some of the concepts in OFBiz.

Some fundamental skill with XML syntax is required, but only as far as is required to create well-formed XML documents.

The book will offer more hand-holding to MS Windows users. Users of other Operating Systems, such as Linux, should naturally be more adept at grasping the concepts laid out using Windows-specific examples and translating those examples into their respective OS's dialects.

## Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: " The `<include-form>` element includes a form named `FirstForm` residing in a file `LearningForms.xml`."

A block of code will be set as follows:

```
<request-map uri="OneForm">
  <response name="success" type="view" value="OneFormScreen"/>
</request-map>
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items will be made bold:

```
server="default-server"  
location="webapp/partymgr"  
base-permission="OFBTOOLS, PARTYMGR"  
mount-point="/partymgr"/>
```

**New terms** and **important words** are introduced in a bold-type font. Words that you see on the screen, in menus or dialog boxes for example, appear in our text like this: "We see two fields **First Name** and **Last Name**, and one **Submit** button."



Warnings or important notes appear in a box like this.



Tips and tricks appear like this.

## Reader Feedback

Feedback from our readers is always welcome. Let us know what you think about this book, what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply drop an email to [feedback@packtpub.com](mailto:feedback@packtpub.com), making sure to mention the book title in the subject of your message.

If there is a book that you need and would like to see us publish, please send us a note in the **SUGGEST A TITLE** form on [www.packtpub.com](http://www.packtpub.com) or email [suggest@packtpub.com](mailto:suggest@packtpub.com).

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on [www.packtpub.com/authors](http://www.packtpub.com/authors).

## Customer Support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.