
Table of Contents

I. Getting Started	1
1. What is Seam	3
1.1. Integrate and Enhance Java EE Frameworks	3
1.2. A Web Frameworks that Understands ORM	4
1.3. Designed for Stateful Web Applications	5
1.4. Web 2.0 Ready	6
1.5. POJO Services via Dependency Bijection	7
1.6. Configuration by Exception	7
1.7. Avoid XML Abuse	7
1.8. Designed for Testing	8
1.9. Great Tools Support	9
1.10. Let's Start Coding!	9
2. Seam Hello World	11
2.1. Create a Data Model	12
2.2. Map the Data Model to a Web Form	13
2.3. Handle Web Events	14
2.4. Better Understand the Seam Programming Model	16
2.4.1. Seam POJO Components	16
2.4.2. Ease of Testing	18
2.4.3. Getter / Setter Based Bijection	18
2.4.4. Avoid Excessive Bijection	19
2.4.5. Page navigation in JSF	20
2.4.6. Access database via the EntityManager	20
2.5. Configuration and Packaging	21
2.5.1. The WAR file	23
2.5.2. The Seam Components JAR	25
2.6. How is this Simple?	26
3. Recommended JSF Enhancements	27
3.1. An Introduction to Facelets	28
3.1.1. Why Facelets?	28
3.1.2. A Facelets Hello World	29
3.1.3. Use Facelets as a Template Engine	31
3.2. Seam JSF Enhancements	34
3.2.1. Seam UI Tags	34
3.2.2. Seam JSF EL Enhancement	35
3.2.3. Seam Filters	37
3.3. Add Facelets and Seam UI Support	37
4. Rapid Application Development Tools	41

4.1. Prerequisites	41
4.2. A Quick Tutorial	42
4.2.1. Setup Seam Gen	42
4.2.2. Generate a Skeleton Application	45
4.2.3. Develop the Application	46
4.2.4. Build and Deploy	46
4.2.5. Run Test Cases	47
4.2.6. Build a Tomcat Target	48
4.3. Work with IDEs	48
4.3.1. NetBeans	48
4.3.2. Eclipse	52
4.4. Reverse Engineer the Database	54
II. Stateful Applications Made Easy	55
5. An Introduction to Stateful Framework	57
5.1. Correct Usage of ORM (Object-Relational Mapping)	57
5.2. Better Performance	59
5.3. Better Browser Navigation Support	60
5.4. Less Memory Leak	61
5.5. High Granularity Component Lifecycle	62
5.6. Reduce Boilerplate Code	63
6. A Simple Stateful Application	67
6.1. Stateful Components	68
6.1.1. Stateful Entity Bean	69
6.1.2. Stateful Session Bean	70
6.1.3. Stateful Component Lifecycle	71
6.1.4. Factory Methods	73
6.2. Page Navigation Flow	73
7. Conversations	77
7.1. The Default Conversation Scope	78
7.2. Long Running Conversation Example	79
7.3. Define a Conversational Component	85
7.4. Start a Conversation	86
7.5. Inside the Conversation	87
7.6. End the Conversation	88
7.7. Links and Buttons	91
7.8. New Frontiers	92
8. Workspaces (a.k.a Concurrent Conversations)	93
8.1. What is a Workspace?	93
8.2. Workspace Switcher	97
8.3. Carry a Conversation across Workspaces	99
9. Transactions	103
9.1. Manage a Transaction	104
9.2. Force a Transaction Rollback	105

9.2.1. Rollback Transaction via Checked Exceptions	105
9.2.2. Rollback Transaction via Return Value	107
9.3. Atomic Conversation (Web Transaction)	107
9.3.1. Manual Flush of the Persistence Context	108
9.3.2. One Transaction per Conversation	109
III. Integrate the Web and Data Components	111
10. Validate Input Data	113
10.1. Form Validation Basics	113
10.2. Validation Annotations on the Entity Bean	116
10.3. Trigger the Validation Action	118
10.4. Display Error Messages on the Web Form	120
10.5. Use JSF Custom Validators	123
11. Clickable Data Tables	125
11.1. Implement Clickable Actions	126
11.2. Display the Clickable Data Table	127
11.3. Seam Data Binding Framework	128
12. Bookmarkable Web Pages (a.k.a RESTful URLs)	131
12.1. Retrieve query parameters in a HTTP GET request	134
12.2. Instantiate a Seam component to handle the query	135
12.3. Start a conversation via a GET request	137
13. The Seam CRUD Application Framework	139
13.1. Data Access Objects (DAOs)	139
13.2. The Declarative Seam DAO Component	140
13.3. A Seam DAO	141
13.4. Queries	142
13.5. Use Hibernate Filters	142
13.6. Customize Error Messages	143
14. Failing Gracefully	145
14.1. Why Not Standard Servlet Error Pages?	146
14.2. Setup the Exception Filter	147
14.3. Annotate Exceptions	147
14.4. Use exceptions.xml to Configure System Exceptions	149
14.5. Debug Information Page	151
14.5.1. The Facelets Debug Page	151
14.5.2. The Seam Debug Page	153
IV. Ajax Support	157
15. Custom and AJAX UI components	159
15.1. AJAX-enabled JSF Components	160
15.2. The Auto-Completion Text Field	161
15.3. AJAX Component Library Packaging for Seam	164
15.4. Other JSF Component Libraries	167
16. Enable Ajax for Existing JSF Components	169
16.1. AJAX Validator Example	169

16.2. Programatic AJAX	172
16.3. AJAX Buttons	175
16.4. AJAX Containers	177
16.5. Other Goodies	178
16.6. Configure Ajax4jsf	178
16.7. Pros and Cons	181
17. Direct JavaScript Integration with Seam Remoting	183
17.1. AJAX Name Validation Example (Reloaded)	183
17.1.1. Server-side Component	184
17.1.2. Trigger a JavaScript Event on the Web Page	185
17.1.3. Make an AJAX Call	186
17.2. The AJAX Progress Bar	188
17.2.1. Seam component	189
17.2.2. Asynchronously access Seam component from JavaScript	191
17.3. Integrate Dojo toolkit	192
17.3.1. Visual Effects	193
17.3.2. Input Widgets	194
V. Integrate Business Processes	201
18. Manage Business Processes	203
18.1. jBPM Basics and Vocabulary	203
18.2. Create a business process	207
18.3. Implement business logic for tasks	211
18.4. Bind data objects in the Process scope	213
18.5. Built-in Seam Components for Task Management	214
18.5.1. The pooledTaskInstanceList Component	215
18.5.2. The pooledTask Component	215
18.5.3. The taskInstanceList Component	215
18.5.4. The taskInstanceListByType Component	216
18.6. Application Users and jBPM Actors	216
18.7. jBPM Libraries and Configuration	218
19. Stateful Pageflows	221
19.1. JSF navigation rules vs. JPD L pageflow	221
19.2. Pageflow and conversations	223
19.3. Pageflow definition	224
19.4. Back button dilemma	225
VI. Testing Seam Applications	227
20. Unit Testing	229
20.1. A Simple TestNG Test Case	231
20.2. Simulate Dependency Bijection	232
20.3. Mock the Database and Transaction	234
20.4. Load the Test Infrastructure	236
21. Integration Testing	239

21.1. An Example Test Script	240
21.2. Build the Data Components from User Input	241
21.3. Invoke UI Event Handler Method	242
21.4. Check the Response	242
VII. Production Deployment	245
22. Java EE 5.0 Deployment	247
22.1. JBoss AS 4.0.5	247
22.2. JBoss AS 5.x	247
22.3. Glassfish	247
23. Seam Without EJB3	251
23.1. Seam POJO With JPA	252
23.1.1. A Seam POJO Example	252
23.1.2. Configuration	253
23.1.3. Packaging	255
23.2. Use Hibernate POJO and API	257
23.2.1. Use the Hibernate API	258
23.2.2. Configuration	259
24. Tomcat Deployment	261
24.1. Package a Seam POJO application for Tomcat	262
24.1.1. Bundle Support JARs	262
24.1.2. Bootstrap the JBoss MicroContainer	263
24.1.3. Configure the Transactional DataSource	263
24.2. Package a Seam EJB3 application for Tomcat	265
24.2.1. Bundle Necessary JARs in the WAR File	266
24.2.2. Bundle Embeddable EJB3 Configuration Files	267
24.2.3. Bootstrap the JBoss MicroContainer	267
24.2.4. Use an alternative data source	268
25. Use a Production Database	271
25.1. Install and Setup the Database	271
25.2. Install Database Driver	272
25.3. Define a DataSource	273
25.4. Configure the Persistence Engine	273
26. Performance tuning and clustering	275
26.1. Tuning performance on a single server	275
26.1.1. Avoid "call by value"	276
26.1.2. JVM options	277
26.1.3. Reduce logging	278
26.1.4. Tune the HTTP thread pool	278
26.1.5. Choose from client or server side state saving	280
26.1.6. Use a production data source	281
26.1.7. Use a second level database cache	281
26.1.8. Use database transactions carefully	283
26.2. Clustering for scalability and failover	283

26.2.1. Sticky session load balancing	284
26.2.2. State replication	285
26.2.3. Failover architectures	285
A. Install and Deploy JBoss AS	287
A.1. JDK 5.0 is Required	287
A.2. Install JBoss AS	287
A.3. Deploy and Run Applications	290
B. Use Example Applications as Templates	293
B.1. A simple template	293
B.1.1. Directories in the template	295
B.1.2. Configuration files	296
B.1.3. The build script	297
B.1.4. The checklist	300
B.2. Extend the template	301
B.2.1. Support Seam UI tags	301
B.2.2. Add Facelets support	301
B.2.3. Add jBPM support	302