ArjunaCore 4.12.0

Development Guide

Building Applications with JBoss Transactions

Mark Little

Jonathan Halliday

Andrew Dinn

Kevin Connor

ArjunaCore 4.12.0 Development Guide
Building Applications with JBoss Transactions
Edition 0

Author	Mark Little	*mlittle@redhat.com*
Author	Jonathan Halliday	*jhallida@redhat.com*
Author	Andrew Dinn	*adinn@redhat.com*
Author	Kevin Connor	*kconnor@redhat.com*
Editor	Misty Stanley-Jones	*misty@redhat.com*

This guide is most relevant to engineers who are responsible for administering JBoss Transactions installations. Although this guide is specifically intended for service developers, it will be useful to anyone who would like to gain an understanding of transactions and how they function.

Preface

1. Prerequisites

This guide assumes a basic familiarity with Java service development and object-oriented programming. A fundamental level of understanding in the following areas will also be useful:

- General understanding of the APIs, components, and objects that are present in Java applications.
- A general understanding of the Windows and UNIX operating systems.

2. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

2.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F1`** to switch to the first virtual terminal. Press **`Ctrl`**+**`Alt`**+**`F7`** to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **`mono-spaced bold`**. For example:

[1] https://fedorahosted.org/liberation-fonts/

> File-related classes include **`filesystem`** for file systems, **`file`** for files, and **`dir`** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

`Mono-spaced Bold Italic` or ***`Proportional Bold Italic`***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **`ssh`** ***`username@domain.name`*** at a shell prompt. If the remote machine is **`example.com`** and your username on that machine is john, type **`ssh john@example.com`**.
>
> The **`mount -o remount`** ***`file-system`*** command remounts the named file system. For example, to remount the **`/home`** file system, the command is **`mount -o remount /home`**.
>
> To see the version of a currently installed package, use the **`rpm -q`** ***`package`*** command. It will return a result as follows: ***`package-version-release`***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

2.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **`mono-spaced roman`** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in **`mono-spaced roman`** but add syntax highlighting as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }
}
```

2.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' won't cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

3. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in the JBoss Issue Tracker: *https://jira.jboss.org/* against the product **Documentation.**

When submitting a bug report, be sure to mention the manual's identifier:
ArjunaCore_Development_Guide

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, please include the section number and some of the surrounding text so we can find it easily.

Overview

1.1. Introduction

1.2. TxCore

The Transaction Engine

1.3. Saving object states

1.4. The object store

1.5. Recovery and persistence

1.6. The life cycle of a Transactional Object for Java

1.7. The concurrency controller

1.8. The transactional protocol engine

1.9. The class hierarchy

Using TxCore

2.1. State management

2.1.1. Object states

2.1.2. The object store

2.1.3. Selecting an object store implementation

2.1.3.1. StateManager

2.1.3.2. Object models

2.1.3.3. Summary

2.1.3.4. Example

2.2. Lock management and concurrency control

2.2.1. Selecting a lock store implementation

2.2.2. LockManager

2.2.3. Locking policy

2.2.4. Object constructor and destructor

Advanced transaction issues with TxCore

3.1. Checking transactions

3.2. Gathering statistics

3.3. Last resource commit optimization (LRCO)

3.4. Nested transactions

3.5. Asynchronously committing a transaction

3.6. Independent top-level transactions

3.7. Transactions within `save_state` and `restore_state` methods

3.8. Garbage collecting objects

3.9. Transaction timeouts

Hints and tips

4.1. General

4.1.1. Using transactions in constructors

4.1.2. `save_state` and `restore_state` methods

4.2. Direct use of StateManager

Tools

5.1. Starting the Transaction Service tools

5.2. Using the performance tool

5.3. Using the JMX Browser

5.3.1. Attributes and Operations

5.4. Using the object store browser

5.4.1. Object state viewers (OSV)

5.5. ObjectStore command-line editors

5.5.1. LogEditor

5.5.2. LogBrowser

Constructing a Transactional Objects for Java application

Appendix A. Object store implementations

A.1. The ObjectStore

A.1.1. Persistent object stores

A.1.2. Common functionality

A.1.3. The shadowing store

A.1.4. No file-level locking

A.1.5. The hashed store

A.1.6. The JDBC store

A.1.7. The cached store

A.1.8. LogStore

Appendix B. Class definitions

B.1. `LockManager`

B.2. `StateManager`

B.3. `InputObjectState`

B.4. `OutputObjectState`

B.5. `InputBuffer`

B.6. `OutputBuffer`

B.7. `Uid`

B.8. `AtomicAction`

Appendix C. Revision History

Revision 0	**Fri Sep 24 2010**	**Dude McPants** *`Dude.McPants@example.com`*

Initial creation of book by publican

Index