ModeShape Artifacts Layout

Walkthrough

The following steps simulate a typical user session. An end result (i.e a layout of file and directories) is then shown.

The user creates a node ('greetings_en') through the UI and uploads content – in this example it's a simple text file (fcrepo4_greetings.txt) with a string ("hello, world!"):

Start Fedora

The default modeshape configuration specifies using a LevelDB backend store.

```
cd fcrepo4/fcrepo-webapp
mvn jetty:run
```

Add Content

```
curl -X POST -H "Content-Type: text/plain" --data-binary "@fcrepo4_greetings.txt" http://localhost:8080/rest
```

Fedora will create a directory "fcrepo4-data" in the current working directory. The default directories found in "fcrepo4-data" will be the following:

```
> ls fcrepo4-data
com.arjuna.ats.arjuna.common.ObjectStoreEnvironmentBean.default.objectStoreDir
com.arjuna.ats.arjuna.objectstore.objectStoreDir
fcrepo.activemq.directory
fcrepo.ispn.repo.cache
```

"fcrepo.ispn.repo.cache" contains the repository metadata as well as the binary content files that are smaller than 4Kb.

Inspecting Generated Data Files

The serialized Fedora nodes can be found in the "fcrepo.ispn.repo.cache/dataFedoraRepository" directory. The files in that directory would look something like this:

```
> ls fcrepo4-data/fcrepo.ispn.repo.cache/dataFedoraRepository/
000003.log
CURRENT
LOCK
LOG
MANIFEST-000002
```

These are the LevelDB cache store files. The generated files contain serialized data about each of the JCR/Fedora nodes. Modeshape depends on Infinispan distributed cache store which in turn uses LevelDB to persist the data to filesystem. To know more about the leveldb files, see: https://leveldb.googlecode.com/svn/trunk/doc/impl.html.

Some key things to note about the files and their contents:

- The serialization is done by the JBoss serialization library, not JDK's native object serialization machinery. For this reason the generated serialized data look different from an ordinary JDK serialized file. JBoss Marshalling can be configured to use custom serialization classes that read and write content in the format of the repository's choosing.
- The data is encoded in Binary JSON (BSON). If the file containing the root node (referencing the node 'greetings_en') is opened up in a hex editor, you would see /u0002 preceding strings (such as "name","key"); /u0004 preceding an array (representing sub-nodes); /u0003 representing the UUID -- in accordance with the BSON spec.
-
```
\u0000\u0000\u0000\u0004children\u0000ü\u0000\u0000\u0000\u00030\u0000<\u0000\u0000\u0000
\u0002key\u0000\u0019\u0000\u0000\u000087a0a8c317f1e7jcr:system\u0000\u0002name\
u0000\u000B\u0000\u0000\u0000jcr:system\u0000\u0000\u00031\u0000X
\u0000\u0000\u0000\u0002key\u00003\u0000\u0000\u000087a0a8c7505d6417c366d9-25bc-4e2d-9e53-
4428fe7b8152\u0000\u0002name\u0000\u0000\u0000\u0000greetings_en\u0000\u0000\u0000\u0003childrenI
```

The binary datastreams larger than 4Kb are stored in the "fcrepo.binary.directory".

Add larger content:

```
> openssl rand -base64 5120 > 5k.file
> curl -X POST -H "Content-Type: text/plain" --data-binary "@5k.file" http://localhost:8080/rest
```

After uploading a binary file (>4Kb), the default directories found in "fcrepo4-data" will be the following:

```
> ls fcrepo4-data
com.arjuna.ats.arjuna.common.ObjectStoreEnvironmentBean.default.objectStoreDir
com.arjuna.ats.arjuna.objectstore.objectStoreDir
fcrepo.activemq.directory
fcrepo.ispn.repo.cache
fcrepo.binary.directory
```

Inspecting the content of the binary directory:

```
> find fcrepo4-data/fcrepo.binary.directory
f5
f5/e5
f5/e5/b6
f5/e5/b6/f5e5b66ef8218602c224ab3099ca0aedcc0b36cb
b5
b5/12
b5/12/1b
b5/12/1b/b5121b0f9270a9a2a2694e5d675b922bec94009e
```

The contents of these files will match the uploaded file and its content-type:

```
> cat fcrepo4-data/fcrepo.binary.directory/f5/e5/b6/f5e5b66ef8218602c224ab3099ca0aedcc0b36cb
Qo7j0VtZy/5dTDk7WB9v4yGRnW2a7H/m1hhdRIa/nEJ562Gg+UpHAN1NmdyzImLv
gyml1ge9UQJjvhHe5KSKepp77CFmSqqRYPakqmGsL6X6dbl51NKYbxoeriemN8TB
pk/N+UKSdC4UtpWBImKCnWJApbCzkA2nf2B/kX6jZGtg5AFDGOB3rXyhIKWNqiX0
Vdcb+BuwBZvjt+y2Ki1YLOkoE44C3Jtorgk7sS5mmU6rzxXSZ7I26S9hfK8Ck7ly
IThy/oRT5+AYhJ7i0r6QSIOTccbmNF0ux4M3oflP3HSoPTsyYOY/henIfN9vrhPN
2DnsyzA4AHn0hwKfhiGa2hA4JIDv2mprNBcCQY4vjX6XY+ExBYvoWCdJySeIspHx
xlDZBZYUyI9ivSmN9muJvlawfcIQTh38ZEkhI6XlQyzoGLjgwcNf6RwznJGX2MFj
mAzBYB3qjMFdpmUUNhet3RAaFLnoSwUyvwJuHaNMl85/SM6WN9qFVIbg6w+sCMTu
fi/KpVzbM5slrWOW1ovJDgiS7ybWo3/Ci4XBHSWNJFJvEYM7QpODlT1XzCc9eKHH
S/sO8AaDBNyS7KTCusJT3jaPnhHgKvqEiSU0yKBMWsaYRiapO3MnFDcjJ3q1ts4I
...

> cat fcrepo4-data/fcrepo.binary.directory/b5/12/1b/b5121b0f9270a9a2a2694e5d675b922bec94009e
text/plain
```

Inspecting ObjectStore Folders

Directories "com.arjuna.ats.arjuna.objectstore.objectStoreDir" and "com.arjuna.ats.arjuna.common.ObjectStoreEnvironmentBean.default.objectStoreDir" are JBoss JTA transaction engine artifacts. The default Fedora Infinispan configuration attempts to find a JBossJTA transaction manager implementation via "org.infinispan.transaction.lookup.GenericTransactionManagerLookup". This configuration uses Arjuna ShadowFileStore as a backend, resulting in several directories within fcrepo4-data such as "object-store" and "object-store-default":

```
|-com.arjuna.ats.arjuna.objectstore.objectStoreDir
|---ShadowNoFileLockStore
|-----defaultStore
|-------Recovery
|---------TransactionStatusManager
|------------0_ffff7f000101_c6cf_56507014_0
|-com.arjuna.ats.arjuna.common.ObjectStoreEnvironmentBean.default.objectStoreDir
|---ShadowNoFileLockStore
|-----defaultStore
```

A detailed description of the artifacts maintained by the JBossJTA implementation is most likely beyond the scope of this document (at least for now).

Infinispan Configuration Options

Depending on the configured Infinispan backend, the directory layout and contents of the binary files would be different. The following sections covers other cache store options.

LevelDB Backend

Currently, the default configuration outputs Fedora data to LevelDB (a fast filesystem based key-value store). When Fedora 4 is started, ModeShape (actually Infinispan and LevelDB in the background) will create several directories on the filesystem. Currently, the directories created are:

1. fcrepo.ispn.binary.cache (binary data)
2. fcrepo.ispn.cache (metadata)
3. fcrepo.ispn.repo.cache (repository)
4. fcrepo.modeshape.index.directory

The layout of files in directories 1-3 is determined by LevelDB. Some of the important files are:

1. File .log holds entries for recent transactions. The relevant API for representing these entries is modeshape-schematics (see, e.g., org.infinispan.schematic.SchematicEntry)
2. File .sst stores these entries when the .log file reaches a size threshold. A new log file is generated.
3. File MANIFEST.x records info about .sst files (among other things).
4. File CURRENT specifies the current MANIFEST file.