Making Open Source Work™

Liferay MongoDB Store

Making Open Source Work™

Table of Contents

1 Introduction

1.1 Overview

Liferay MongoDB Store connects Liferay Document library with MongoDB. With the use of this framework, we can configure document library to store documents in MongoDB and also enhance the performance and scalability benefits of MongoDB.

1.1 Overview

This component is dependent on Liferay Portal and MongoDB. We have tested this component for the following versions of Liferay Portal and MongoDB:

- Liferay Portal EE 6.1 GA2
- MongoDB 2.0.4

1.3 Technology Details

This component is developed based on following technologies, frameworks and libraries:

- Liferay Portal
- MongoDB
- MongoDB Java Driver

1.4 References

- http://www.mongodb.org/display/DOCS/Home

2 User Guide

2.1 Overview

Liferay has built in support to store documents of document library to database, file system, Amazon Simple Storage Service (S3), JCR based repository and CMIS supported repository.

Liferay MongoDB Store enables Liferay to use MongoDB as a document repository. MongoDB is a highly scalable NoSQL database. By using MongoDB for storing documents, we can leverage benefits of MongoDB like scalability and auto sharing capabilities. These, in turn, allow us to store terabytes of document data in Liferay document library.

2.2 Features

2.2.1 Adding support of MongoDB

This is the basic feature of this framework. It connects Liferay document library and MongoDB. MongoDB has a component called GridFS. GridFS allows us to store large files. GridFS divides large files into chunks. By connecting document library with MongoDB, we can leverage all the features of MongoDB.

2.2.2 Replication, Clustering and Sharding documents

MongoDB is designed to work with huge data stores. In order to achieve that, it has implemented features like clustering of multiple MongoDB nodes, replication on many MongoDB nodes and sharing capabilities. By using this framework, we can leverage all these benefits for Document Library.

2.2.3 Configurable

Liferay MongoDB Store is highly configurable. We can enable it by adding a property in portal-ext. properties file. We can also configure the location of MongoDB database using portal-ext.properties. The following sections will provide more details on the same.

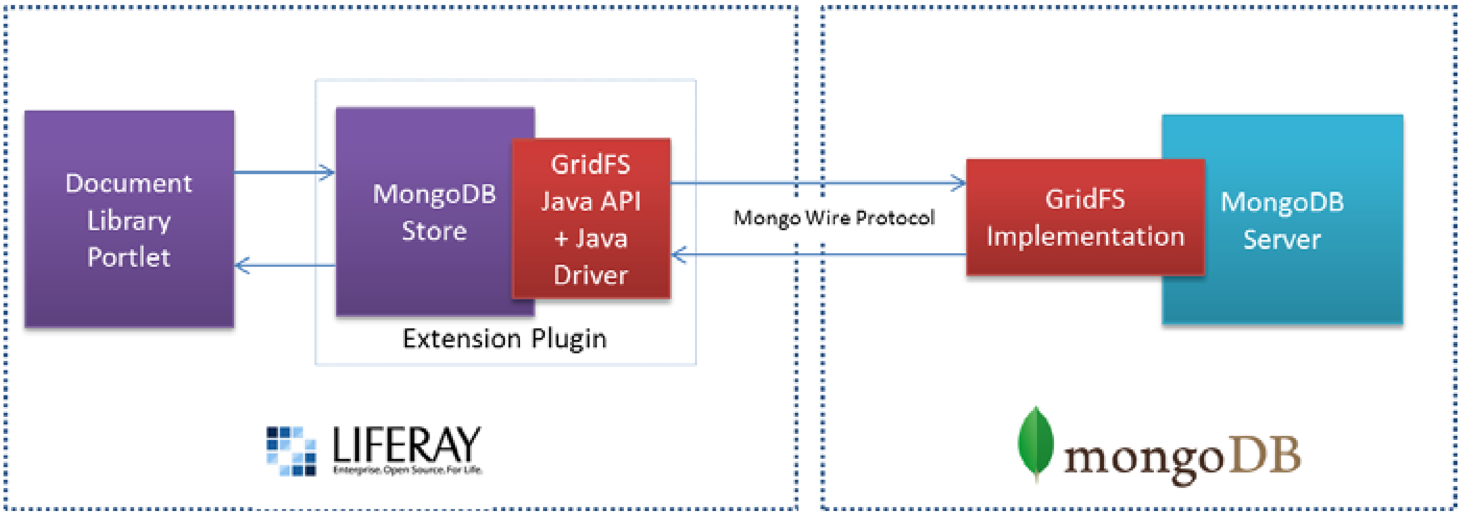
3 Technical Details

3.1 Software Components

This component is developed using a Liferay Extension plugin. In order to deploy and test this component we need the following software components:

- Liferay Portal EE 6.1 GA2
- MongoDB 2.0.4

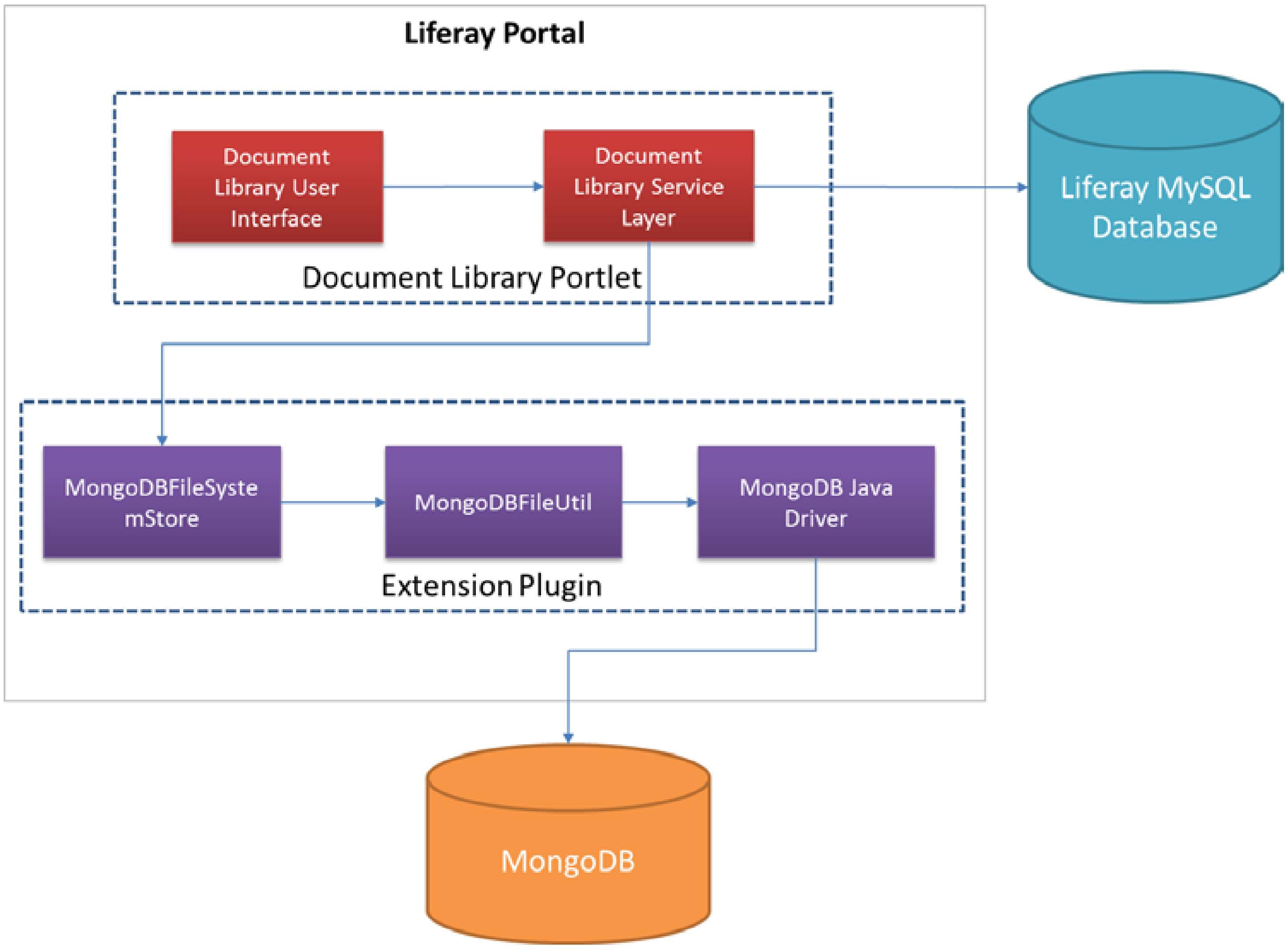
3.2 Architecture Diagram



As shown in the diagram, the framework provides MongoDB store by using Liferay extension plugin. Using Liferay portal configuration file, document library portlet can be configured to use MongoDB store. Once configured, all document upload or download requests from document library portlet will be delegated to MongoDB store. MongoDB store will use MongoDB java driver and GridFS java API to store or retrieve documents from MongoDB. The java driver uses Mongo Wire Protocol to connect with the MongoDB server.

3.3 Component Design

This component is developed using an extension plugin, as we wanted to develop a new type of document store. The following diagram explains how these components interact with MongoDB to store and retrieve documents.



As shown in above diagram there are three key components in Liferay MongoDB Store.

1. MongoDBFileSystemStore

This class extends to com.liferay.portlet.document library.store.BaseStore which is the parent class of all types of document stores. Liferay 6.1 provides a configuration property to hook any type of document store who implements com.liferay.portlet.documentlibrary.store.Store. BaseStore is an abstract class implements this interface.

In this class we have implemented all the methods which are needed by the Store interface. MongoDB File System Store delegates calls to MongoDB File Util class. MongoDB FileUtil class connects with MongoDB through MongoDB java driver.

The Store implementation can be configured by overriding dl.store.impl property in portal-ext.properties file.

2.MongoDBFileUtil

This class actually connects with MongoDB Java driver. It connects with MongoDB instance configured in portal configuration file and performs operations to store and retrieve documents from MongoDB.

3.MongoDB Java Driver (mongo-2.7.3.jar)

MongoDB ships with a Java Diver to connect to MongoDB and GridFS. We have used this driver inthis component.

4 Build Process

We developed this component as part of Liferay Plugin SDK. In order to build this component you need to follow following steps.

1. Download and configure Liferay 6.1 EE Plugin SDK.
2. Now unzip mongodb-connector-src.zip file provided with the component in ext folder of your pluginsdk.
3. Now from command prompt, navigate to <PLUGIN_SDK_DIR>/ext/MongoDB Store-ext directory.
4. Run following command.
   ant war
5. You will find MongoDBStore-ext.war file in <PLUGIN_SDK_DIR>/dist directory.

5 Configuration & Installation Details

In this section we have tried to explain steps to configure all these softwares:

5.1 MongoDB Setup and Configuration

5.1.1 Linux Environment Configuration

5.1.1.1 Overview

This section describes the basic configuration required for the Linux environment, so that user can work easily with the environment.

5.1.1.2 Create New Linux User (cignex)

MongoDB simply runs on the non-root user. So if you have the root user rights and you do not have any other user available, then create the non-root user by performing the following steps:

- Run the command adduser <user name>.i.e. adduser cignex and follow the given straight forward steps to add an new user in the system.

Note: - For all operations use cignex user unless any other user specified for the particular operation

5.1.1.3 HTTP Proxy Configuration

To download any binary installation on Linux environment, we must have to specify the http proxy used by the system. To set the proxy in Linux environment, follow the below given steps:

- Login to the Linux environment using cignex user.
- Open the /etc/wgetrc file in text editor.
- Add http_proxy=http://<proxy server>:<proxy port> at end of the file
- Which means <proxy server> specifies the proxy server to use.
- <proxy port> specify the proxy server port.
- Add use_proxy=on at end of the file. This specifies that use the proxy while downloading any file using wget command.
- Save and Exit the editor.

5.1.1.4 JDK Installation

To set up JDK follow the below given steps.

- Log in to the Linux environment using root user.
- Run the command apt-get update to update repository. Apt stands for Ubuntu's
- AdvancedPackaging Tool (APT).
- Run the command apt-get install openjdk-6-jdk to install the Open JDK.
- Log out from root user and re-login to the Linux environment using cignex user.
- To set the JAVA_HOME environment variable, go to the home of Linux environment. i.e. cd ~
- Open .bashrc in text editor and add the following line at end of the file.
- Export JAVA_HOME=<JDK Installation path>.

5.1.1.5 Create Data Directories

For storing the data, create the directories in separate partition where enough disk space is available. We assume that separate partition is accessible using /Data. Follow these directions to create.

- Log in to the Linux environment using the root user.
- For the MongoDB create a MongoDB directory. And it is accessible using /Data/MongoDB.
- Then provide read write access to all users using command:chmod 777 Data/MongoDB.

5.1.1.6 Create Log File

For storing the log, create the file in separate partition where enough disk space is available. We assume that separate partition is accessible using /Data.

- Log in to the Linux environment using the root user.
- Create the mongoDB.log file. It is accessible using /Data/mongoDB.log.
- Also provide full read write access using chmod 777 /Data/mongoDB.log.

5.1.1.5 Create Data Directories

5.2.1 Download

To download the MongoDB, run the below given command from <SERVER_HOME>/servers in this case it is /opt/mongoDB/ directory in Linux environment. We are using MongoDB 2.0.4.

For 32 Bit server:

wget http://downloads.mongodb.org/linux/mongodb-linux-i686-1.6.4.tgz

For 64 Bit server:

wget http://downloads.mongodb.org/linux/mongodb-linux-x86_64-2.0.4.tgz

The above command will download the MongoDB package in the directory, from where the command was executed.

5.2.1 Download

- To unpack the gz file run gunzip <.gz file, in our case mongodb-linux-x86_64-2.0.4.tgz. This will extract mongodb-linux-x86_64-2.0.4.tgz in to mongodb-linux-x86_64-2.0.4.tar file.
- To unpack the tar file run tar xf <.tar file, in our case mongodb-linux-x86_64-2.0.4.tar. This will extract mongodb-linux-x86_64-2.0.4.tar in to mongodb-linux-x86_64-2.0.4.tar.gz directory.

5.2.3 Configuration

5.2.3.1 Setup user and database on MongoDB server

- Startup Mongodb server using following command <MONGODB_HOME>/bin/mongod
- Now connect to mongodb server using MongoDB shell by the following command <MONGODB_HOME>/bin/mongo
- In the mongo shell prompt type following command to create admin user and dlfileentrydatabase use dlfileentrydb.addUser("admin", "sysadmin")
- Now shutdown the MongoDB server using the following command in mongo shell use admindb.shutdownServer()

5.2.3.2 Create MongoDB Configuration file

For the MongoDB we have to configure various files in the <MONGODB_HOME> directory. Here, <MONGODB_HOME> is the MongoDB installation base directory. i.e
/home/mongodb/mongodb-linux-x86_64-2.0.4

<MONGODB_HOME>/mongoDB.conf

Edit this file to configure MongoDB store information like dbpath, logpath, port number and fork flag.
Add the properties in the format below dbpath = /Data/MongoDB/mongoDbData
logpath = /home/mongodb/mongodb-linux-x86_64-2.0.4/mongodb.log
logappend =
true port =
27017 fork =
true
auth = true

5.2.3.3 Start MongoDB server using new configuration

We can start by using the following command.

```
<MONGODB_HOME>/bin/mongod --config <MONGODB_HOME>/mongoDB.conf \
```
Or
Create shell script with the above command and run that script.

After starting the server please check the generated log file from <MONGODB_HOME> directory, (as we did logfile generation path in above configuration) to check whether server started properly or is there any error in the server start up.

5.2.3.4 Stop MongoDB Server

We can stop by using the following command.

```
<MONGODB_HOME>/bin/mongo
>use admin
>db.shutdownServer()
```

5.3 Liferay Configuration

5.3.1 Installation

Download Liferay 6.1 EE GA2 addition and place it to some directory i.e. in /opt/liferaymongodb/liferay-portal-6.1.10-ee-ga2/ directory in linux environment.

5.3.2 Startup and License Installation

- Goto <LIFERAY_HOME>/tomcat-7.0.25/ and execute blow command to start Liferay./bin/startup.sh
- Now copy your license file to <LIFERAY_HOME>/deploy directory.

5.3.3 Component Installation

- Copy MongoDBStore-ext.war file to <LIFERAY_HOME>/deploy.
- After deployment edit <LIFERAY_HOME>/tomcat-7.0.25/webapps/ROOT/WEB-IN F/classes/ portal-ext.properties and configure following properties. dl.store.impl=com.cignexdatamat ics.portlet.documentli brary.store.MongoDBFileSystemStore
- dl.store.mongoDB.host=[MongoDB server IP] dl.store.mongoDB.port=27017 dl.store.mongoDB. database= dlfileentry
- dl.store.mongoDB.username=[username of mongodb server's dlfileentry data base]
- dl.store.mongoDB.password=[password of mongodb server's dlfileentry data base]
- Restart the tomcat server as this component is developed using extension plugin.