The OpenBrr project

The aim of this project is to develop a hosted data aggregation application for all F/LOSS projects.

The project consists of two components – OpenBrrClient and OpenBrrServer. The OpenBrrClient is a flex-based component. Per Tony, this component needs to be changed to an open source technology. Therefore this document does not describe this component further.

Following are the main steps of the OpenBrrServer:

1. The crawler crawls the FlossMole DB in order to identify the files that match a specified input.
2. The crawler then downloads all the identified files to the local file system.
3. The downloaded files are in the compressed form. The server then unzips the files and sends them for parsing.
4. The parser reads the files into memory, parses files to fetch the required fields into memory and then stores them in the local DB.

Building the "OpenBrr" project

Pre-Requisites

- Java SE 6.x
- Apache ANT 1.7
- Flex SDK version 3.3
- Sun GlassFish server/ Apache Tomcat Server
- Eclipse IDE - Recommended

Setting up the Environment

- Add ANT binaries to command line path
- Copy <Flex_SDK_HOME>/ant/lib/flexTasks.jar to <ANT_HOME>/lib
- Download Glassfish Server / Use Apache Tomcat Server
- Use the ANT targets to build, deploy, and run the project

ANT Target Reference

Target	Description
init	An internal target used to initialize the properties
compilejava	Compile the Java Code
compileflex	Compile the Flex (ActionScript and MXML) code
deploystaging	Deploy source to staging area. This is a wrapper in "deploy-ror-

	staging", "deployjava-staging", and "deploy-flex-staging"
deployrorstaging	This is used to deploy only the Ruby on Rails code to the staging area. This is useful when ROR code is updated
deployjavastaging	This is used to deploy only the Java code to the staging area. This is useful when Java code is updated
deployflexstaging	This is used to deploy only the Flex code to the staging area. This is useful when Flex code is updated
deploystructure	This target is used internally to copy the more static directory and files to the deploy area.
start-gf	Start the Glass Fish server
stop-gf	Stop the Glass Fish server
sdeploysolr	Add the SOLR web application to the staging server
sdeployobrr	Add the Obrr web application to the staging server
clean	Clean all intermediate files / directories including compiled files

Checkout and Build From Source

1. To download source code for Openbrr, please follow download instructions at http://github.com/OpenBrrCMU/OpenBrr.
2. Copy build.properties.sample to build.properties. Configure the following path in the build.properties file
    - project.home - location of the source files
    - staging.dir - directory to host the staging server
    - production.dir - directory of production server
    - flex.sdk.home - location of the Flex SDK files
3. Copy glassfish.properties.sample to glassfish.properties. Configure the following properties in glassfish.properties file. These are glassfish server specific properties. For documentation please reference Glassfish Server documentation.
    - DOMAIN
    - ADMIN_PORT
    - ADMIN_USER
    - PASSWORD_FILE
    - DOMAIN_DIR
    - GF_HOME
4. Run "ant deploy-staging" to compile and create the staging server.
5. Run "ant start-gf" to start the glassfish server.
6. For the first time and every time "SOLR" configurations are changed, run "antsdeploy-solr". This will deploy the SOLR web-application.
7. Run "ant sdeploy-obrr" to run the Rails application.
8. Set up the DATA_FOLDER in the 'openbrr.collector.flossmole.FlossmoleConstants.java' file to a directory on your local server

Running the OpenBrrServer

To run the OpenBrrServer through the above steps locate the 'FlossmoleCollector.java' file under 'org.openbrr.collector.flossmole' package. Run this file as the main class. The following steps are executed when this file is run:

1. This class first calls the crawler that crawls the FlossMole DB in order to identify the files that match the specified input in the PROJECT_CODE.java file.
2. Crawler (Crawler.java) then downloads all the identified files to the local file system. The files are downloaded to the 'UNPROCESSED_FOLDER' specified in 'FlossmoleConstants.java' file.
3. Then the FlossmoleDataProcessor.java is called. This class unzips the files and then parses them. The files are unzipped into the 'UNPROCESSED_FOLDER' specified in 'FlossmoleConstants.java' file.
4. When parsing, it reads the files into memory, parses files to fetch the required fields into memory. The parsed files are placed in the 'PROCESSED_FOLDER' specified in 'FlossmoleConstants.java' file
5. The final step is to store the parsed files in the local DB.

All the key files on the OpenBrr2Server module have appropriate comments to describe this process. This procedure will be clearer as you run the server module. It is recommended that the server module be run in a step-by-step manner initially to understand the flow better.

Error encountered

Currently the OpenBrrServer successfully goes through all the four steps described above but it fails at the step 5 when it needs to store the parsed files in the local DB. You can connect the OpenBrrServer with a MYSQL DB by specifying the DB connection details in the PersistenceUtil.java file. Thereafter the components mentioned in the 'Next Steps' section need to be developed in order to complete the project.

Next Steps

There are two main steps that need to be looked into in order to complete this project:

1. The schema for the DB needs to be developed. This schema should be consistent with the one used in the files in the 'openbrr.core.data' and 'org.openbrr.collector.flossmole.data' packages. This is the only step left in order to completely execute the OpenBrrServer. Once the correct schema is developed in a MySQL DB, the OpenBrrServer should be able to store the parsed files in this newly developed DB.

2. An OpenBrrClient needs to be developed. The current OpenBrrClient is flex based and this needs to be changed. The OpenBrrClient should provide a user interface to get the user input to allow them to search the F/LOSS projects. It should then connect with the OpenBrrServer in order to complete the search successfully.

References

- Sun Glassfish Server
  - http://www.sun.com/software/products/glassfish_portfolio
  - https://glassfish.dev.java.net/
- Flex SDK
  - http://opensource.adobe.com/wiki/display/flexsdk/Flex+SDK
- Project hosting at GitHub
  - The project is hosted at: http://github.com/OpenBrrCMU/OpenBrr
- To browse source code:
  - http://github.com/OpenBrrCMU/OpenBrr