Feedr Developer Manual

Welcome to the Feedlr Developer Manual. This document is meant to act as a guide for new developers interested in contributing to the Feedlr project. As this document will not hold the answers to everything we encourage you to refer to a member of the developer group for any further questions.

Installation

1. The source code can be found in our git repository at https://github.com/blueliine/feedlr/. After downloading the project import the whole folder by "Android Project from Existing Code" found under *File -> New -> Other -> Android* in Eclipse and choose the feedlr folder. After importing the projects right click on the feedlrTest and select properties and in *Java Build Path -> Projects* press *Add* and select the Feedlr project.

   For more information on git usage refer to http://git-scm.com/documentation.

2. To develop for Android you will also need the Android SDK. For download and installation instructions, go to http://developer.android.com/sdk/.

Application structure

The application package structure is designed mainly after class type. A package for Android Activities, one for Android Services etc. For a visual representation of the application structure, refer to the *Architecture Specification* document.

Packages

The .client package holds classes and helper classes for handling client authorization and request actions. In the case of Twitter the classes are TwitterHelper for handling all sorts of request to the Twitter API and TwitterAuthHelper for authorization requests.

The .parser package contains classes for parsing the JSON response from the client requests into Java Objects to be used in the app directly or saved to the database.

The .database package contains classes for handling the database.

The .model package contains simple model classes defining our base object types (Feed, FeedItem, User etc.). The model classes are mainly used by the .parser and the .database package when parsing responses from the client REST APIs and extracting data from the applications database.

The .service package contains an Android Service DataService and a helper class. The DataService operates in the background handling all client request, parsing and saving to the database on request from an Activity. When a request is parsed and saved to the database a Broadcast to the Activity will be made reporting that new data is available.

The .ui package contains classes extending Android View and Android Fragment associated with the graphical user interface. Note that most of the GUI is defined in the resource XML-files.

The .activity and .adapter packages contains Android Activities and Android Adapters. Together these are responsible for responding to user actions, making calls to the DataService and supplying and updating the views with data from the database.

Release Procedure

Before a release there is a few things to do.

1. Create a release branch
2. Add release content
3. Add additional branches included in the release
4. Make sure that the project builds and runs properly
5. Run tests (Junit tests, Acceptance tests and EMMA)
6. Merge with master
7. Create Release notes
8. Create Test Report
9. Create Test Report Results
10. Create Coverage Report
11. Add Release notes, Test Report, Test Report Results and Coverage Report to /dist/ folder.
12. If needed, update the documentation in /docs/
13. Create .apk
14. Tag the last commit with release version

Generating coverage report with EMMA

To generate a test coverage report with EMMA, make sure all your projects are updated (including library projects). See http://developer.android.com/tools/projects/projects-cmdline.html#UpdatingAProject for more information.

Run the following command from the main project folder:

```
> ant emma debug install
```

Followed by the following command from the test project folder:

```
> ant emma debug install test
```

The report file coverage.html can be found in the test project's bin folder.