News Streams User Manual
May 13-31

Sections:

1) Front End(Website) User Guidelines

a) How to setup

User setup for the website is fairly straightforward. You would place the sources files for the front end onto a working web server, and then you can access the frontend by using any web browser and entering the url for the webserver.

b). How to use

Once the Front End is setup properly you can use the website like you would any other website. There will be the main feed which displays your articles with next and previous buttons to cycle through articles. When you click on a story a new tab will be opened with that Story contents from the original sources. The category list on the left side will allow you to choose a category by clicking on it and then having related articles displayed in the news feed. There are buttons for redirecting to both login pages and custom source pages. The custom source page is toggle buttons with the name of sources on them. By toggling or pressing down a button a source will be used, while un-toggling a button will exclude a source from being used. The login page is a standard login page which will take text input of a username and password and match it to one in the database.There are navigation buttons to every other page on each page.

2) Database User Guidelines

a) How to set up

User setup for the database is relatively simple assuming that a suitable MySQL database is already available. One must edit the preferences xml file to reflect the location of your MySQL database. The location entry should be filled out with the web address of the database and along with the port number on which it is listening; typically this is port 3306. Additionally one must change the user and password entries to reflect the username and password which is to be used for accessing the MySQL database. All jsp files used by the user front end must be similarly updated.

b)How to Use

Once the database is properly setup there is no user interaction with it, as all its functions are handled internally by the program.

3) Backend User Guidelines

a) Acquiring Source Code

The code for this project is stored on a Google Code svn located at https://code.google.com/p/senior-design-prototyping/. It is open source so anybody can import the project if you wish to make changes to it you will need to be added to the list of contributors. To be added contact Jamison Bradley at jsbradley523@gmail.com.

b) Deploying the Project

The project should be jarred and then deployed like that, the main method for the project is found in the class RSSMonitor.

c) Preferences and Their Meanings

The project has a file called preferences.xml in the manager package and this contains preference that control how parts of the system work.

edit_distance_coe - Is the coefficient used by the QuotationScorer class to determine how much weight the edit distance algorithm should be given in the score that is returned. This value should be between 0 and 1. Also edit_distance_coe + longest_common_substring_coe should equal 1. Experimental results that were carried out during our development show that this algorithm performs better and should probably be given the higher coefficient between the two.

longest_common_substring_coe - Is the coefficient used by the QuotationScorer class to determine how much weight the longest common substring algorithm should be given in the score that is returned. This value should be between 0 and 1. Also edit_distance_coe + longest_common_substring_coe should equal 1.

error_log_location - This is the file that errors are logged to. It should be set to an absolute location on the computer that the system is going to be running on.

*special note for next three values - The system can be ran without using a database and just using local storage in this case nothing would need to be provided in the database fields.

database_host - This is the location where the database is being hosted.

database_username - This is the username that will be used when logging into the database.

database_password - This is the password used when logging in that corresponds to the database_user value.

maximum_date_match - This is to determine the maximum amount of time that can elapse between when two articles are published to be considered part of the same story.

time_between_rss_checks_ms - This is the amount of time that should elapse before the RSSMonitor class checks the RSS feeds again for new links. The time should be entered in milliseconds, it is recommended that the time always stay above at least 10 minutes or the system could choke.

threshold_score_for_match - In the ArticleAggregator class you will find the three scorers being used to determine the overall score for a match between two articles. This value is the overall score that needs to be achieved in order for an article to be considered a match with another one. The value that it is currently set at was determined by running experimental with the goal being to maximize correct matches and minimize incorrect matches.

date_scorer_coe - This is a coefficient that modifies the score coming out the DateScorer object in the AritcleAggregator class, increasing this coefficient will give the DateScorer more control over determining matches decreasing it will give it less control. The current value in use was determined experimentally.

quotations_scorer_coe - This is a coefficient that modifies the score coming out the QuotationsScorer object in the AritcleAggregator class, increasing this coefficient will give the QuotationsScorer more control over determining matches decreasing it will give it less control. The current value in use was determined experimentally.

keyword_scorer_coe - This is a coefficient that modifies the score coming out the KeywordScorer object in the AritcleAggregator class, increasing this coefficient will give the KeywordScorer more control over determining matches decreasing it will give it less control. The current value in use was determined experimentally.

rss_feeds - These are the RSS feeds that are being monitored for new links.

D) Populating Database

It is recommended that when you run the RSS monitor for the first time to populate the database that you set the time_between_rss_checks_ms value to a very high value like a couple of hours. Then add some print statements to the RSSMonitor main method to keep track of progress. When it is done kill the program and set the value of time_between_rss_checks_ms to a lower value. This is needed because the first time you run the main method to populate the database it will be pulling in a huge amount of articles and it could choke the system if you try to run it with a normal time between checks due to the first check not being done when the second one starts.

E) Adding New Scorers

Create a new class and implement the IScorer interface and implement the score method so that it returns a score of 100 for a perfect match between two articles and a 0 for a worst possible match and a score between 0 and 100 for matches in between perfect and worst possible. Then edit the ArticleAggregation class so that it takes into account the new scorer you created, and add an entry in the preferences for the new scorers coefficient.

G) Adding New Sources

Create a new class that implements the IParser interface and implement the parse method so that it returns an Article object with the required fields. Look at other parsers to see where logging errors should take place. When done modify the SourceParser class to use your new Parser.

H) Manager Class

This class should always be used to access any instance of the ILogger, IDatabase, or IPreferences interfaces. This class allows you to write your own implementations of these interfaces and integrate them into the code simply by changing the return value in the Manager class, since it is used to get ahold of those implementations one change here will allow the new implementation to plug in to the code without anything else needing to change.