SE 3XA3: Software Requirements Specification
Module Interface Specification

Team #, Team 3
Erin Varey, Vareye
Nik Novak, Novakn
Joel Straatman, Straatjc

November 13, 2016

Contents

List of Tables

Table 1: **Revision History**

Date	**Version**	**Notes**
Date 1	1.0	Notes
Date 2	1.1	Notes

1 Introduction

Module Interface Specification is a necessary component in the design of Rather. The decomposition of the program through modules allows for various changes to be anticipated by abstracting various processes and required components. Many components specific to a certain API are subject to frequent change, especially social media related API's. Below is the Module Interface Specification for the Chrome extension Rather.

2 Popup Module

Popup.js is the isolated javascript module that interacts with popup.html. It controls the main interface with the user and is in charge of passing messages to the page context javascript files. Methods from this class are always invoked upon user interaction.

Interface (public entities)

2.1 send_tags()

Sends the set of tags input by the user on the chrome extensions main HTML. Communicates through message passing across Chromes security border.

2.2 send_blockwords()

Sends the set of blockwords input by the user on the chrome extensions main HTML. Communicates through message passing across Chromes security border.

2.3 send_settings()

Sends a settings packet from Settings.js. Used to determine program functionality based on options stored in packet. Settings are sent on any update of user settings.

2.4 on_click()

Called when an item in the HTML is interacted with. Subsequently calls on_update(event) with the appropriate event identifier.

2.5 on_update(event)

Called anytime the user interacts with the main interfaceand causes a notable change. The event variable determines the occuring event and on_update performs the appropriate message passing to the main page-context program.

3 Page Module

Page is the main controller script, used to control information flow and general program structure. It is at the top od the module hierarchy, and passes mesasges with the main chrome extension HTML.

Interface (public entities)

3.1 receive_tags()

Requests the set of tags input by the user on the chrome extensions main HTML. Communicates through message passing across Chromes security border.

3.2 receive_blockwords()

Requests the set of blockwords input by the user on the chrome extensions main HTML. Communicates through message passing across Chromes security border.

3.3 receive_settings()

Requests a settings packet from Settings.js. Used to determine program functionality based on options stored in packet.

3.4 load_tags()

Called when tags are updated by on_update() of popup.js. Searches and re-injects all posts using subsequent inject() methods determined by the service_visited variable

3.5 save_tags()

On program successful termination, save_tags is called to ensure tag words are properly loaded at next runtime

3.6 load_blockwords()

Requests the set of words to filter, stored by previous runtime and updates the user's blockwords text box

3.7 determine_service()

Determines which API to utilize based on current tab's address bar; and calling the subsequent service API's inject method. If no known service is recognized, Default-API's inject method is called.

3.8 load_filter()

Loads the set of names to exclude from filtering, leaving implementation to the Filter-Feature.js's load_names()

3.9 save_filter()

Saves the set of names to exclude from filtering, leaving implementation to the Filter-Feature.js's save_names()

4 Filter-Feature Module

Filter-Feature.js is the module for adding the implementation of filtering certain names from post-injecting.

Interface (public entities)

4.1 load_names()

Loads the list of names to be disregarded during injection. The list is stored in a CSV formatted text file FilterNames.csv.

4.2 save_names()

Saves the list of names to be disregarded during injection. The list is stored in a CSV formatted text file FilterNames.csv, overwritten each time.

4.3 clear_all()

Removes all names from the filter, as well as from offline storage by deletion of Filter-Names.csv

5 Twitter-API Module

API for implementation of Twitter specific post filtering. In the event that Twitter's HTML structure changes, this module is the only change required.

Interface (public entities)

5.1 search_posts(keywords)

Called when Page.js determines it is on a page. Returns the list of all posts with content matching the keywords provided.

5.2 inject(settings_packet)

Calls search_posts(keywords) to determine the list of posts to be replaced, then based on settings in settings_packet and filter-feature, will inject content determined by Content.js to replace the post content.

6 Facebook-API Module

API for implementation of Facebook specific post filtering. In the event that Facebook's HTML structure changes, this module is the only change required.
Interface (public entities)

6.1 search_posts(keywords)

Called when Page.js determines it is on a page. Returns the list of all posts with content matching the keywords provided.

6.2 inject(settings_packet)

Calls search_posts(keywords) to determine the list of posts to be replaced, then based on settings in settings_packet and filter-feature, will inject content determined by Content.js to replace the post content.

7 Default-API Module

Default-API for content injection. In the event that a service is not recognized, every child node of an HTML document containing Javscript text_fields and containing keywords will be removed.
Interface (public entities)

7.1 inject(settings_packet)

Based on settings in settings_packet and filter-feature, will inject content determined by Content.js to replace any text node containing tags from within the settings_packet

8 Detect-Website Module

This Module determines the service visited by the user in the current tab, and maps it to a specific service_ID. Other modules may use this identifier in determining which implementation to use for post content editing.
Interface (public entities)

8.1 detect()

The main method for detection of the site or service being visited. This module uses the page's HTML structure to determine the service. If the service is undeterminable through HTML structure, it calls upon parse_address() to determine the site. If parse_address and HTML content detection fail, Default-API's identifier is returned. Otherwise the site specific identifier is returned.

8.2 parse_address()

Retrieves the current tab's address bar content and attempt to determine the service or site being visited based on its string content. Returns the appropriate identifier.

9 Settings Module

Settings Module for storing any user settings selected by the user.

9.1 receive_settings()

Returns a settings packet for determination of program behaviour based on user settings. Packet is null if no settings are present. Calls all subsequent load_* methods in this module to retrieve the settings information.

9.2 load_tags()

Specific implementation of retrieving tags input by the user out to a text file, previously stored. May be called at any time by Page.js

9.3 save_tags()

Specific implementation of storing tags input by the user out to a text file, text file is overwritten each time. May be called at any time by Page.js

9.4 load_blockwords()

Specific implementation of retrieving blockwords input by the user out to a text file, previously stored. May be called at any time by Page.js

9.5 save_blockwords()

Specific implementation of storing blockwords input by the user out to a text file. May be called at any time by Page.js

10 Content Module

The Content module determines which content will be replacing a post marked for injection. Typical replacement content includes: images. Highly subject to change and additions such as text replacements.

10.1 image(keyword)

Retrieves an image based on var keyword, for post injection. Implementation is left to Image.js's image_retrieve(keyword)

11 Image Module

Image module abstracts the implementation of retrieving images for Content Module. Uses RSS feed to retrieve images based on keywords

11.1 init()

Initializes RSS feed image retrieval API

11.2 image(keyword)

Retrieves a list of images based on var keyword, for post injection, using an instagram RSS Feed. Returns said lsit with a mazimum size of N images, determined in settings.