

COMS E6156 Topics in Software Engineering

Final Report

Localization Tool for Developers and Translators

Jong Ho Lee, jl4421@columbia.edu

As briefly discussed in the midterm paper, software localization process has gained importance as it became easier to release software to different local market. However, the localization remains as one of less explored topic both in academia and industry despite the rising importance. With the recent efforts of platform developers including Apple, Google, and Microsoft, internationalization and localization of platform-specific application has become easier than the past with well-written documents and API/SDK provided by each vendor. However, localizing web application remains as a difficult task due to the lack of standardized process which is offered for vendor-specific applications. And because there are various implementations of internationalization tool, attempting to standardize localization process for web application would be practically impossible. Thus, to aid developers in internationalizing web applications and translators with localizing the web application in context aware manner, I'm delivering a localization tool for web application as the final project for the course. The project is delivered as an open source project via Github as recommended by Professor Kaiser (https://github.com/faketraveler/webapp_l10n). Instructions on how to use this program is provided in the README.md at the Github repository.

The project is primarily written in JavaScript, with the help of HTML and CSS to render the changes in the user interface. The software used in this project include jQuery for DOM manipulation [1] and bower for dependency management [2]. Additionally, the sample modal in

the program, which is also used in the demo, uses JavaScript and CSS from Bootstrap for effects and styling [3]. The program will use these two libraries but is not extending them. As a JavaScript program running on the client-side, the tool will not have requirements other than a ES5 compatible browser such as Mozilla Firefox or Chromium-based browser like Google Chrome. To run the program, the developer will import the JavaScript package with Bower, which also handles the dependencies of the program. This way, the developers can call the functions of the program in the initializer/constructor or in front-end frameworks such as vue.js or backbone.js via AMD with require.js or webpack. Or the program and its dependencies could be loaded in the html with script tag. The program is intended for the developers, mostly front-end developers, to prepare the web application for translation and translators to do actual translation.

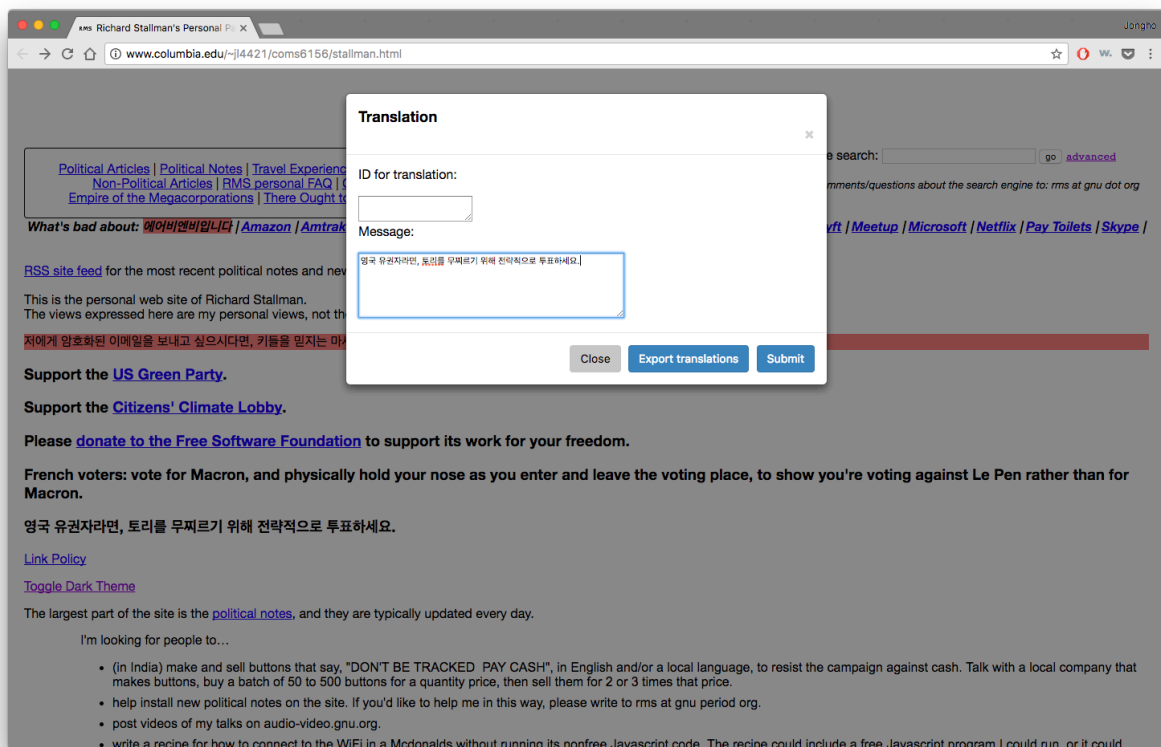
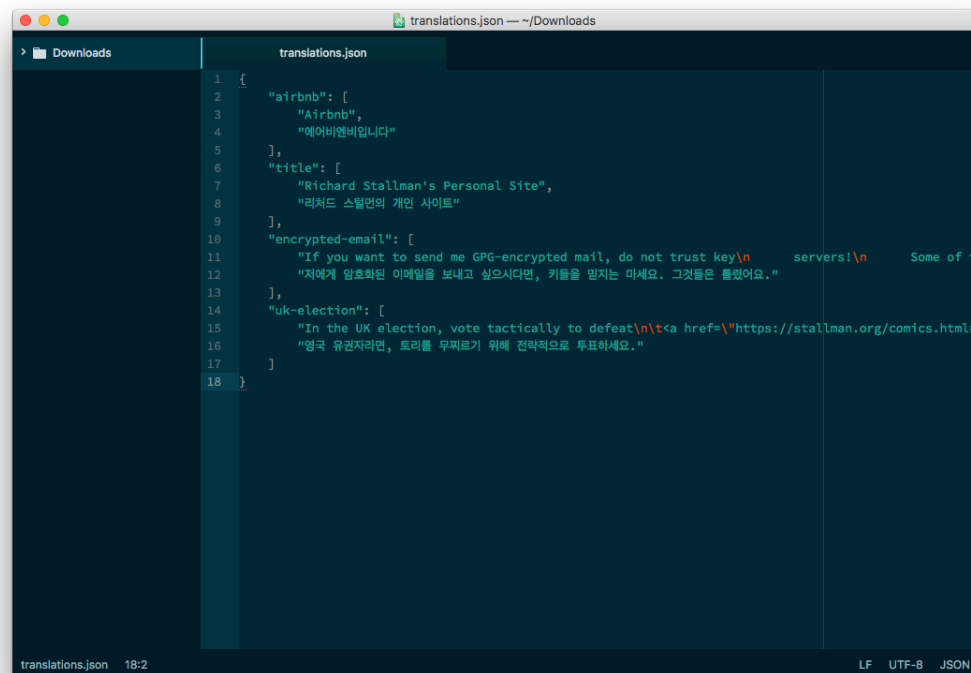


Figure 1. Demo of the program with Richard Stallman's Personal Website

The program provides following functions to help the developers to internationalize their web applications and allow the translators to perform context-aware translation of each page, rather than performing simple string translation. First, the tool allows all elements with messages to be edited when it is clicked. On click, a modal will pop up, allowing the translator or developer to change the content of targeted DOM element with new message. The modal has 2 textboxes, one for the title or identifier for the target message, and the other for the actual translation of the message. The title or identifier will be used as the key in a JSON object, allowing JSON export of the translations. In addition to this modal, the tool highlights the elements with a significant change either in width or height with noticeable but transparent color with alpha value. The default threshold for significant change is 20 percent increase or decrease in either width or height, which could be customized the developer. After the translation process is complete, the translator can click on a button to export all the translation in the page as a JSON



```
1 {
2   "airbnb": [
3     "Airbnb",
4     "에어비앤비입니다"
5   ],
6   "title": [
7     "Richard Stallman's Personal Site",
8     "리처드 스탈만의 개인 사이트"
9   ],
10  "encrypted-email": [
11    "If you want to send me GPG-encrypted mail, do not trust key servers! Some of 1
12    "저에게 암호화된 이메일을 보내고 싶으시다면, 키들을 믿지는 마세요. 그것들은 틀렸어요."
13  ],
14  "uk-election": [
15    "In the UK election, vote tactically to defeat\n\t
```

Figure 2. Translations exported as a JSON file

file. As briefly mentioned earlier, the JSON file will have the title or identifier as the key, and an array of messages as the value, where the first element is the original message and the second element is the translated message.

Because this project was extended from my midterm paper covering the same topic, I there weren't too many things which I learned specifically learned in the development process, as I learned many things throughout the research process during writing the midterm paper. One of the few things which I learned then was that in many smaller organizations, the localization process resembled my experience, where they hire individual translators and give them a text file to translate, and they often do not consider the UI as a factor when they translate the messages. It also could have been so because I had relatively enough experience in web application development, so I was aware of potential problems or obstacles that I might face during the development process. Nonetheless, there were new findings and learnings after the development process as well. In an attempt to evaluate the usefulness of this program, I reached out to a number of developers to use this program for localization, and provide feedback on this program and localization in general. Total of three developers took the time to use and provide feedbacks for this program, and the feedbacks were received in person and over the phone. I've also attempted to circulate a survey in different international developer communities, mainly Korean developer communities in Facebook because I had easier access to them, but it was not successful because I was not able to provide an easy installation process when I first circulated the survey. In these feedbacks I received, there were few things I learned. One of it was a technical feedback, something I had missed in the development process. One of the developers who tested the program pointed out that the program does not well with html `<a>` tags, since the program binds browser's "click" event to show the modal and "click" event on `<a>` tag cannot be

overridden. Thus, messages in `<a>` tags needed to be changed to `` or `<p>` in order for them to be "translatable". I've attempted to fix this issue, but did not have enough time in between receiving the feedback and submitting the final report to do so. Another finding from the feedback received from the developers was that this program seemed to be an effective solution for many web application. All of the three developers who tested this program commented that this program would be very effective in localizing web application, especially because the dimension change alert idea in the program. One of the three developers also commented that he would in fact use this application in his localization process, and extend it so he could integrate it with his server so that the actual translation file, in his case .po file, can be updated as he localizes each page. It was interesting to find that intuitively the idea appeared as an effective tool assisting in the localization process.

In the process of finalizing the program, there a feature which has been removed after receiving recommendations in class and piazza or found difficulties in implementing them. The removed function was auto-translation of the strings in the web application using API from machine translation programs such as Google Translate or Bing Translation. However, with the feedbacks I received during my presentation, it was pointed out that machine translation, despite their improvements with neural network algorithm, tend to be inaccurate when the sentences are longer. And because the machine translation feature was intended for developers to provide some sense of what the localized outcome would be, I determined that it could be misleading for developers if the translated messages from the machine translation is very different when compare to legitimate translation in terms of length.

The difficulties in localizing software stem from the localization and translation process. Usually, localization, technically preparing the software for designated locale, is done by

developer, while translation, preparing the contents of software for designated locale, is done by translator who is not very knowledgeable of the original user interface. For both web application and platform-specific application, the translator receives only the messages which needs to be translated. Because not all translators are familiar with software localizing process, many translations end up being translated without the consideration of user interface. As a result, the translated web application may have several broken components in its user interface after localization. This project aims to solve this specific problem and provide tools both for developers and translators. With the tool, developers can learn which components of the user interface may need additional attention by translator, and translator may translate the messages directly from the pages of web application. This program would provide better translation experience for the translators and better i18n/l10n experience for the developers.

References

- [1] “What is jQuery?” *jQuery*. Accessed April 25, 2017. <https://jquery.com/>
- [2] “Bower - a package manager for the web” *Bower*. Accessed May 9, 2017. <https://bower.io/>
- [3] “Customize and download” *Bootstrap*. Accessed May 9, 2017. <http://getbootstrap.com/customize/>