Web Localization in Implio

Name: Kesing Huang
Class: Front 18

Supervisor: Lars Forsberg & Karl West
Besedo Front End Developer: Ada Ge

Summary

Besedo is a company witch has a content moderation tools named Implio. Since Besedo has clients from all around the world, to have different language on the product, that help moderators from different country and region is gaining a lot of interest from both Besedo and the client.

Localization in the web development word usually called “l10n”, According to W3C[1]: "Localization refers to the **adaptation** of a product, application or document content to meet the language, cultural and other requirements of a specific target market”. And in this project, I will try to do the localization in English, the language Implio currently using.

Internationalization also known as i18n, after complete the Localization, then the website will be possible to have all the hard coded string in one place, which will make the Internationalization much easier.

In this report, I will focus on explaining the localization in AngularJS framework.

[1] https://www.w3.org/International/questions/qa-i18n

1.Introduction

1.1 Background:

Although 59.6% of the website's content are in English(https://w3techs.com/), there is, for example "roughly 917 million of whom speak Mandarin"(https://www.babbel.com/). We can say it will bring benefit to all the website if it can provide the local language to user from all around the world.

Implio is a content moderation product from company Besedo, which has clients from all around the world. We first received the request from one of the moderater: "As a moderator that moderates in a different language than English I would like Implio to be translated to my language so that it is easier and goes faster for me to moderate."
Therefore it would have given a great benefit for the company to have informative and improve efficiency for the moderator from all around the world.

2.Method

2.1 Objectives and criteria:

The first step would be to do the localization in English that would be worked in both AngularJS and React, and the future plan would be translation into other languages.

2.2 Problem Analyse:

- First step is the localization, we need to make sure all the hard coded string is collect to the same JSON file, and all the strings that show on the site is coming from the same source. Which is time consuming.
- Second step will be translate all the string in the JSON file.
- Third step, we should investigate how the translation might be effect Implio UI.
- Fourth step, we need to implement an UI function to let the user change between different language.

2.3 Goal

To make sure Implio website is localized in English as much as possible.

3.Theory

AngularJS[2]:

AngularJs is one of the early Javascript framework, which was created in 2009 by two developers Misko Hevery and Adam Abrons, the idea was to minimise their code in a project, and later under Google's support it developed to became one of the beloved Javascript framework among developer at that time.[3]
Implio is built in AngularJS, when I stated the project, our front-end lead have already built up the environment for this project to continue, which is an Angular module that I would talk about in the next section.

Angular translate module[4]

Is a module that can use in Angular framework to make the localization easier. The front-end lead in Besedo created an Angular translation filter and a JSON file to collect all the string in one place, and through the translation filter to show the string in the whole site.

JSON(JavaScript Object Notation)[5]

"is a lightweight data-interchange format."[6] And it is easy for both machine and human to understand. It shown as a Javascript object and appear as a collection with name and value.
In the following paragraph, I would explain in detail about the working process.

---

[2] https://angularjs.org/

[3] https://andrewaustin.com/an-overview-of-angularjs-for-managers/

[4] https://angular-translate.github.io/

[5] https://www.json.org/json-en.html

[6] https://www.json.org/json-en.html

1. To implement in XML(E**x**tensible **M**arkup **L**anguage) file.
   So “Try Again” is the hard coded string, we collect this string to JSON file and use the key “AUTOMATION.DECISION_RULES.BTN.TRY_AGAIN”, to access this string from JSON file. The 't' is a filter service from a filter file called “t.js”.

Before

```
<div class="filter-view content">
  <div class="loading-error" ng-show="vm.errorLoadingFilter">
    <h2>{{vm.errorText}}</h2>
    <a class="btn eva-btn-primary" ng-click="vm.reloadFilter()">Try Again</a>
  </div>

  <div ng-if="!vm.errorLoadingFilter">
```

After

```
<div class="filter-view content">
  <div class="loading-error" ng-show="vm.errorLoadingFilter">
    <h2>{{vm.errorText}}</h2>
    <a class="btn eva-btn-primary" ng-click="vm.reloadFilter()">{{'AUTOMATION.DECISION_RULES.BTN.TRY_AGAIN'| t}}</a>
  </div>

  <div ng-if="!vm.errorLoadingFilter">
```

2. To implement in CSS file
   First give a attribute name data-content, and use the JSON key to find the right string.

```
data-content="{{'MANUAL.QUEUE_LIST_PAGE_BODY.ADD_QUEUE_ICON_TOOLTIP' | t}}" ></div>
```

   Second in CSS ::after state, link the content from the attribute

```
    &:after {
      font-size: 14px;
      position: absolute;
      opacity: 0;
      -webkit-transition: all 0.2s;
      transition: all 0.2s;
      content: attr(data-content);
      width: 130px;
      font-size: 13px;
      top: 12px;
      left: -115px;
    }
```

3. To implement in Angular Javascript file:
   First inject the filter service in the controller.

```
  .controller('filterViewCtrl', FilterViewCtrl);

function FilterViewCtrl($scope,
  $stateParams,
  $controller,
  ApiFilters,
  UserPermissionManager,
  tFilter) {
```

Second use it as a function in the Javascript

```
tFilter('SETTING.TEAM_MEMBER.CREATE_INVITE.FAILURE_MESSAGE')
```

4. Using logic in JSON file
   First we would assign the COUNT valuable to vm.numberOfAds in javascript file, and we would use the module function called message format, without it would just show the JSON object value as it written.

```
text: tFilter('SETTING.RIGHT_TO_BE_FORGOTTEN.DELETE_USER_DATA_MODAL.TEXT1',
  {COUNT: vm.numberOfAds, USER_ID: vm.userId}, 'messageformat')
```

   In the JSON file if the number(COUNT) is equal 1 then it would appear “1 item” and if is other number than 1 it would appear ex. “ 6 items”. So it basically put an if statement in JSON.

```
{COUNT, plural, =1{1 item} other{{COUNT} items}}
```

Way to Organise JSON:

There is already rules about how the localization JSON keys should be named.

1. Basic rule from Besedo: "Naming convention for translations keys: A key should describe the string's location in the page, and it's type (label, placeholder, link ..etc). For example: `NAV.PRIMARY.HELP_MENU.API_DOCUMENTATION_LINK` describes a link to API documentation, in the help menu of the primary nav bar."
2. When there is a new window/modal in the browser, we would make a child object.
   ex. the upload file is a different section

```
"CSV_IMPORTER": {
  "TITLE": "Import your sample data from CSV file",
  "UPLOAD_FILE": {
    "HEADER": "Upload File",
    "DRAG_DROP_TEXT": "Drag and Drop a CSV file here",
    "SELECT_FILE_BTN": "Or select a file to upload",
    "NEXT_STEP_BTN": "Next Step",
    "NEED_HELP_TEXT": "Need help?",
```

3. Error message we would collect in same section
   ex. (see next page)

```
"API_INTEGRATION": {
  "TITLE": "API Integration",
  "TRY_AGAIN_BUTTON": "Try Again",
  "API_KEY_TEXT": "API key",
  "WEBHOOK_TEXT": "Webhook URL",
  "COPY_BUTTON": "Copy",
  "SAVE_BUTTON": "Save",
  "API_DOC_TEXT": "API Documentation",
  "API_DOC_LINK": "The API documentation can be found in the following <a
  "ERROR": {
    "FAIL_TO_LOAD_DETAILS": "An error occurred when loading the details",
    "INVALID_URL": "You must provide a valid HTTPS URL.",
    "SAVE_FAILURE": "Failed to update the api integration URL."
  },
```

Debugging:

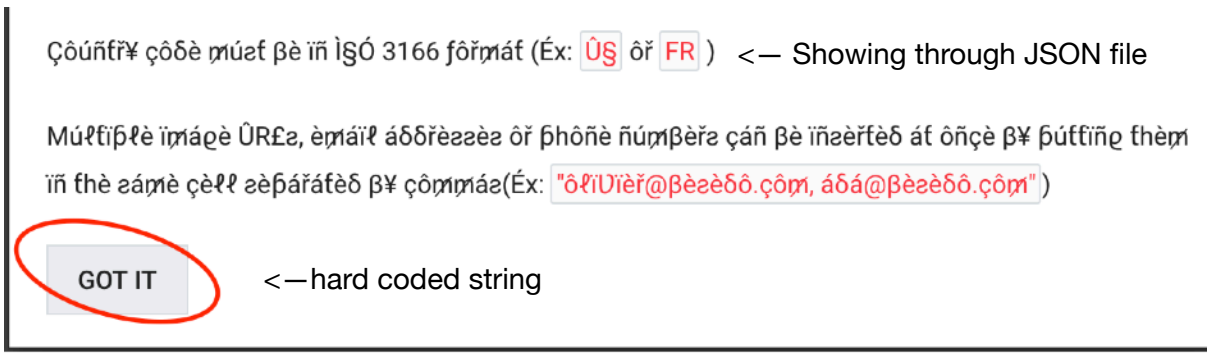
React intl universal pseudo converter npm package.[7]
The package convert the localize json file string into pseudo character to detect the hard code string.
1. Use comment in terminal: $ react-intl-universal-pseudo-converter create -f ./src/translations/en.json -o ./src/translations/fake.json
(this will convert the en.json value to pseudo character as picture bellow)

```
{
  "Company_label": "Tïƒℓè ƒôř ƭhè çôɱβáñ¥",
  "navigation": {
    "menu_open": "Óβèñ ƭhè ɱèñú",
    "account_login": "ƒôϱ ïñ ƭô ƭhè áççôúñƭ",
    "account_logout": "ƒôϱ ôúƭ ôƒ ƭhè áççôúñƭ"
  },
  "about_page": {
    "title": "Âβôúƭ βáϱè",
    "Subtitle": "§ôɱèƭhïñϱ áβôúƭ ƭhè áβôúƭ βáϱè"
  }
}
```

Source:(https://www.npmjs.com/package/react-intl-universal-pseudo-converter)

2. Change the source path in config.js from en.json to fake.json

How it looks like in the browser :



Tools:

1. Visual Studio Code[8]: The tool for code editing that comes with many useful plugging that help the code editing made easy. The most useful plugging is Code Spell Checker[9] for spelling and ESLint[10] for the Javascript style and prevent code style error from the start.

2. Git[11]: File/text version control, which help our individual and cooperation made easy, we can share our code through git and test others code. Mostly use Terminal to control.

3. Sourcetree[12]: Since Git from the beginning is control over CLI(Control Line Interface) in the terminal. This is a GUI (Graphical User Interface) for Git, which can be easier to see the change and different branch you are working on.

4. GitLab[13]: The tool for cooperation and share our in GUI (Graphical User Interface), which sync with Jira so we can check what is the ticket requirement. We mostly use it for code review to comment to other's code and improvement. And detect the merge conflict in an early stage, and it runs pipeline and to make sure all the test is error-free.

5. Jira[14]: Tool for Agile working process, where we can have information about the work progress of ourselves and others. Also use this tool in the spring planning.

6. Terminal[15]: Use CLI(Control Line Interface) to run, test our code and use git to control the different version of the code.
7. Slack[16]: Since Besedo development teem is sitting in different corner of the world, so this is a tool for us to share idea, code, bug and information. During the Covid-19 time is extra import.

---

[8] https://code.visualstudio.com/

[9] https://marketplace.visualstudio.com/items?itemName=streetsidesoftware.code-spell-checker

[10] https://eslint.org/

[11] https://git-scm.com/

[12]https://www.sourcetreeapp.com/

[13] https://about.gitlab.com/

[14] https://www.atlassian.com/software/jira

[15] https://en.wikipedia.org/wiki/Computer_terminal

[16] https://slack.com/

Discussion

Although this might seems like an easy job to do, it is extremely time-consuming. When doing code review about the localization, it also takes a lot of time to check if all the string is appearing correctly. And many times, it might be the tooltip of a small icon. The smallest mistake would make the site looks unprofessional. As well as different decision about the JSON key naming is also will decide the future developer would be easy to understand the code and the string behind the key just by looking at it.

Conclusion

The process of Localization implementation gives a new insight about Front-end web development. For user the translation will be only one click away, behind that one click is a lot of different accumulated knowledge.
The project is not complete finish, soon in the future I would try to do the Internationalization and since the localization is in place, the translation/Internationalization would be effortless than the Localization.

Reference

Project website:

Implio: https://app.implio.com/

Articles:

1. W3TECHS *Usage statistics of content languages for websites* https://w3techs.com/technologies/overview/content_language 06/03/2020
2. ANGULAR JS https://angularjs.org/
3. Andrew Austin *An Overview of AngularJS for Managers* https://andrewaustin.com/an-overview-of-angularjs-for-managers/
4. ANGULAR TRANSLATE https://angular-translate.github.io/
5. JSON *Introducing JSON* https://www.json.org/json-en.html

Tools:

1. Visual Studio Code https://code.visualstudio.com/
2. Code Spell Checker https://marketplace.visualstudio.com/items?itemName=streetsidesoftware.code-spell-checker
3. ESLint https://eslint.org/
4. GIt https://git-scm.com/
5. SourceTree https://www.sourcetreeapp.com/

6. GitLab https://about.gitlab.com/
7. Jira https://www.atlassian.com/software/jira
8. Mac Terminal https://en.wikipedia.org/wiki/Computer_terminal
9. Slack https://slack.com/

Link that can see the visual changes:
https://friendly-sinoussi-ea7ef8.netlify.app/
Github page:
https://github.com/SvampK/Localization