Internship Report

Development of a multi-language solution for a web application

Name: Viktoriya NIKOLOVA
Studying for: BSc(Hons) Computing
Location: LIRIS
Supervisor: Mr. Alain MILLE
Duration: 08/04 to 13/06/2014

Acknowledgement

First I would like to thank Mr. Pierre-Antoine Champin for giving me the opportunity to do an internship within the research centre LIRIS. Although quite short, for me this was a great experience I can learn from. It helped me to explore my skills and increased my interest in web development.
Special thanks to Ms. Fatma Derbel and Mr. Alain Mille for being so accommodating and understanding.
I would also like to thank Ms. Anne Corrigan and the staff at IUT for helping me during my mobility period. I appreciate their patience and help.

Abstract

The key to having successful and fully functional web applications is in their communication with the user. It is of no surprise that human/machine interaction is a popular topic of research and development, as is the main goal at the research centre for Information Technologies LIRIS, where I had the honour to spend my first training period.

A solution has been proposed for the internationalisation and customization of web applications in general. Taking into account web standards and the back-end and front-end architecture of web development, a method for providing a multi-language web interface has been planned from scratch and explained in details in this report.

Table of Contents

1. Introduction

As a full time student at University of Abertay Dundee I have decided to undertake a semester of my second year studies at IUT Lyon 1. The degree towards I am currently studying is a Bachelor of Science in Computing. The course at the IUT was quite challenging for me for a few reasons: Firstly, the programming languages and development tools I was using at the IUT were new for me, hence I had to quickly integrate not only with my new living environment but also academically. This was very beneficial for me as at the end I could compare what I've learnt with what I already new and find a connection between the two. Since I find myself more interested in the area of web development and design, I was happy to be assigned the position of a trainee in web development in a research centre situated on campus - LIRIS.
This report is a description of my 10 week internship carried out as a compulsory component of the course at IUT. In the following chapter details of the activities of my team are given. Afterwards, I explain my role and tasks as a trainee and give specific technical details about my main tasks. Finally, a conclusion is drawn from the experience.

1.1. Context of Project

The team SILEX (Supporting Interaction and Learning by Experience) conducts research in the field of knowledge construction, user assistance, system adaptation to the user, and usage analysis by the user. The topic of the SILEX team is about the interaction between human and machine. They design new methods and solutions to successful knowledge transfer between users and web technology.

One of the projects that my team works on ,COAT, consists of the design and implementation of a Trace-Base Management System for the web platform ClarolineConnect.

The platform provides highly functional tools for the development of personal learning environments or educational devices. Thus, users can manage, store, share and disseminate information while having high level monitoring.
Traces of user activity are collected and stored for further exploration and analysis, which contributes to the improvement of the learning experience. The so-called modelled trace (M-Trace) is proposed as a digital object with specified properties, well defined models, and operators with a precise semantics.
While the implementation of this project requires complex back-end programming and data management, it is of equally high importance that the delivered data to the user is understandable and accessible through good design and communication, or front-end programming.

1.2. Context of Internship

I have been assigned the task to provide translation(internationalisation) for the Trace collecting web extension Trace-Me and the Assistant of Samo-TraceMe, and suggest a solution to translating and dynamically editing the content of a web page in general. As previously mentioned, the M-Trace object contains information of the user traces (action history on the ClarolineConnect platform) which allows a detailed preview of each trace, containing information, such as date and URL. This is where I fit in. For a better user experience, an option to choose a language of preference had to be added to the pages displaying and managing the trace view. This is way of establishing good communication with the user, while making it accessible for a wider audience. Furthermore, the solution had to be extended and modified in order to comply with any web page written in HTML.
More details on my weekly tasks can be found in Appendix A. Implementations will be discussed further ahead in the report.

1.3. Technical Aspect

For the completion of the tasks the following web languages and scripts have been used: HTML for the construction of the layout, CSS for the design of the layout, JavaScript for dynamic functioning and JSON for data storage.

Kango - a cross-browser extension platform was used to provide compatibility to the extension for all major browsers. The Kango framework includes useful features. Along with the options to write background scripts that process large data in background and browser management, the kango storage has been found useful to the Trace-Me extension for reasons including storing user made changes, such as the language preference.

2. Solutions

2.1. Trace-Me

During the first half of my internship I was working on the translation of the web plug-in Trace-Me. The extension consists of a pop-up window which allows the user to choose when traces are collected by activating/deactivating the application. Notifications are therefore used to notify the user whether traces are collected while they are browsing. Additional option page is available in order to manually add a new trace.

As seen on the figure 1 and 2 (see page *), a list of three flags of the countries of the languages are displayed on the upper right corner of both pop-up and option windows. A single click changes the language settings of the extension, storing the preference in local storage for future reference.

I have used two different strategies for the translation of the windows. However, a general solution has been developed and will be further explained in the report.

For better display in the po-up I have decided to include the image buttons as another header cell, whereas in the option window they are stored inside a list table.

The solution is specific for the extension as all text values are collected and replaced using id recognition of predefined id's.

By clicking a flag, a function according to the desired language is called. What the function does, is create a JSON object which contains the new text values to be displayed and replace the previous ones by calling the tags by their id's.

2.2. Translate and Edit Module

While the first solution is simple and functional, it is not developer-friendly and does not give any room for further user and admin configurations.
A more general solution has been proposed for the following scenario:

A user would like to view a trace collection that includes many details. However, they are not happy with the long or vague names of the actions, therefore would prefer if they could be shorter but still understandable. They would want their default language to be German, but display some of the text as desired. On the other hand, an administrator has spotted a few disparities along the translation of the text and therefore would like to quickly change them permanently.

My main task was to find a solution for this scenario that would also work for other web applications. What I was striving for is to develop a small extension (in the form of a folder) that can be built into any web source and provide the functionality to edit text data on the client side for the users and on server side for the administrators and give the ability to translate the same data, without affecting its accessibility levels.

I found myself sifting through large amounts of information trying to find the best solution for this problem, therefore some aspects of it are not yet fully developed. In the next chapter you will find more deeper explanation of how I went about suggesting the above solution.

3. Discussion

3.1. Overview

The Translate and Edit application had been planned to consist of two parts- front-end and back-end development. The front-end is the part of the web that you can see and interact with (e.g. Client-side programming). While front-end code interacts with the user in real time, the back-end interacts with a server to return user ready results. The front-end is a combination of HTML,CSS and JavaScript coding. By using JavaScript, modifications of the design of a web page can be made immediately, however only temporary and visible only by the user.

Normally the user would not have rights to modify web content dynamically on the server side.

Logically, administrators are the ones who deal with back-end modification of databases for example, as they often contain sensitive data which should not be available to see or modify by the general public. Back-end programming languages include PHP, Python, Ruby and others.

As I have minimal experience with back-end programming, I have initially focused on the front-end development of the Translate and Edit module. However, if a developer were to extend its functionality, they would be able to reuse code that manages user edits for their benefit.

On figure 3 (see page *) it can be seen clearly seen how front-end and back-end development differentiate and where is their common point. Further explained you will find out more about each of the components in the above-described module.

3.2. JSON

The database in this case is the JSON library file, stored on the server-side and parsed upon request. As JSON was used primarily in the Trace-Me web extension to store data, I have decided it would be a good idea to use it as a method of internationalisation for the Assistant of Trace-Me, mainly because of the simplicity of adding data, which is an important part of the module.
By creating an array of objects, each containing default text from the Assistant and it's translations, I was able to compare every text attribute on the page with the existing ones in the library and replace them appropriately.

3.3. HTML

What I had to take in mind prior to starting the project was accessibility issues and web standards.
I had written a strict XHTML file that contains two sections - one list (<ul>) section for the flag buttons and another (<div>) for the modification buttons.
One major concern of web accessibility is the use of images.It is considered best practice to add "alt" and "title" attributes for users who cannot distinct images. For example the image of the German flag has a title "Deutsch" and alt attribute set to "german flag".
The lang attribute is also set as english (lang="en") in order to inform the browser of the default human language of the script, which is essential for the proper reading of the web page by certain technologies for the disabled.

3.4. Style/ CSS

The main styling is stored in an external spreadsheet, although the HTML DOM style object has also been used to change some settings while the JavaScript is being loaded. For example the Save button for the edit module has set visibility to "hidden" in the external stylesheet, but the property changes to "visible" when the edit button is clicked to avoid potential confusion.

Another method I have used to change properties inside JS is the jQuery .css() method. I have found it to be effective in changing background and border properties of objects.

3.5. JavaScript

All functionality of the modules has been programmed in JavaScript, including jQuery and AJAX. jQuery is a fast and small JavaScript library that offers many useful features that make event handling among other things much simpler with an easy-to-use API that works across a multitude of browsers.

AJAX, though not another programming language or library is a way of using existing standards. It is the art of exchanging data with a server and updating parts of a web page, without the need to reload the entire web page.

As AJAX was already used to dynamically load data for the Trace-Me Assistant, I have found certain AJAX event handlers in jQuery to be useful for my application.
I used the jQuery.ajax() handler which performs an asynchronous HTTP (ajax) request. The request is sent to the translation library (or the json file) leading to the creation of a JSON object upon success. The object, stored in a variable called libData will be further used to refer to text values in the application.
The idea of the edit module is that once the user decides they want to change the screen text of a web page, all of the text values would become editable by a single button click. For this purpose, the following actions need to occur:

- ✓ Collect all HTML tags on the web page and store them in variables;

- ✓ Perform a check if each tag contains text;
- ✓ Highlight the tags with found text values upon mouse hover;
- ✓ Wrap the text inside a text box and allow modification;
- ✓ Distinct the modified text from the unmodified (e.g. Outline the text box);

The next step of the edit process would be to save the changes by clicking on the 'Save' button. What happens is the following:

- ✓ Input values are collected and stored in arrays;
- ✓ A new JSON object is created;
- ✓ Whenever the user changes a text value, it is stored locally inside a JSON object;
- ✓ Display new text values on reload;

Optionally, the user should be able to reset all text nodes in their original state. That is made possible by adding another button called 'Default', which deletes the object in local storage.

Some of the main JavaScript code can be find attached as Appendix B on page *. Explanations of what each bit of code does are provided.

The functionality of the back-end office modifications would be exactly the same with the only difference that the changes will be made permanent. It would be possible to reset default(old) values, however that would mean not only changing data but adding more to the database. This could bring up some complications, the most obvious one being processing time.

3.6. Problems and Suggestions

All in all, the methods proposed in the solution are quite general, however not fully completed. There is plenty of room left for improvement. For instance, while accessibility issues have been handled, some JavaScript functions remain accessible only at user click which could be difficult for those that do not use a mouse output. I would suggest that tab properties are added to all text that allows modification in order to make sifting through them easier and on click event handlers are extended to work on key press events as well.

I have struggled to find the best practice of comparing strings. Not all web applications are programmed perfectly with text that is always enclosed in tags, or properly spelled, which means a comparison is not full and might result in a glitch.

4. Conclusion

In a nutshell, this internship has been an excellent and rewarding experience. I can conclude that there have been a lot I've learnt from my work at the research centre. Needless to say, the technical aspects of the work I've done are not flawless and could be improved provided enough time.

As someone with no prior experience in JavaScript whatsoever I believe my time spent in research and discovering new languages was well worth it and contributed to finding an acceptable solution to an important aspect of web design and development. Two main things that I've learned the importance of are time-management skills and self-motivation. Although I have often stumbled upon these problems at University, they had to be approached differently in a working environment.

I have yet to complete another two years of studies, in order to achieve a bachelor degree in Computing (with Honours). Working with web development languages has increased my interest in them, hence prompting me to transfer to the Web Design and Development course at my university.

List of figures

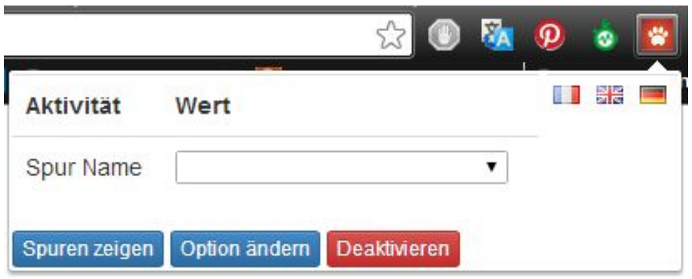
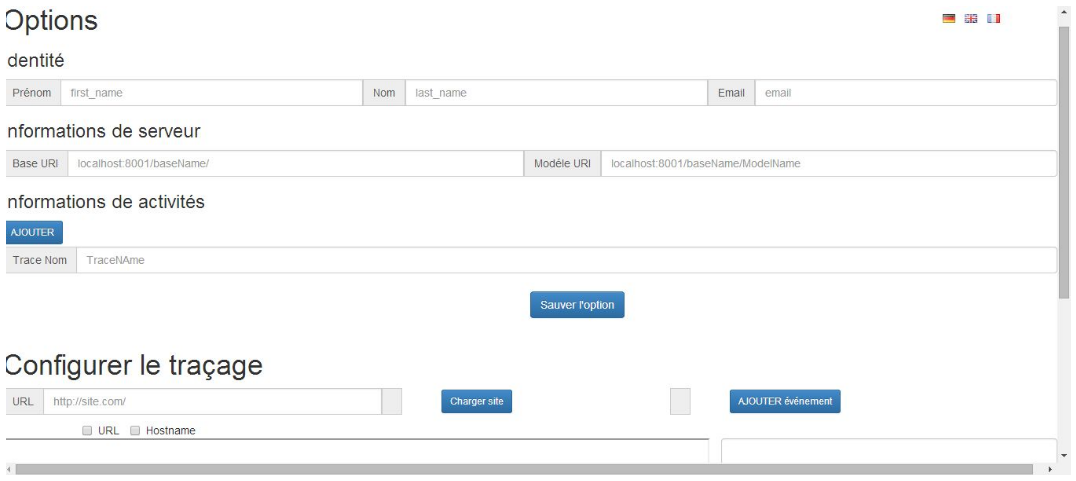


Fig. 1 - Trace-Me web extension pop-up window

Fig.2 - Option page window for Trace-Me



Options

dentité

Prénom first_name | Nom last_name | Email email

nformations de serveur

Base URI localhost:8001/baseName/ | Modéle URI localhost:8001/baseName/ModelName

nformations de activités

AJOUTER

Trace Nom TraceNAme

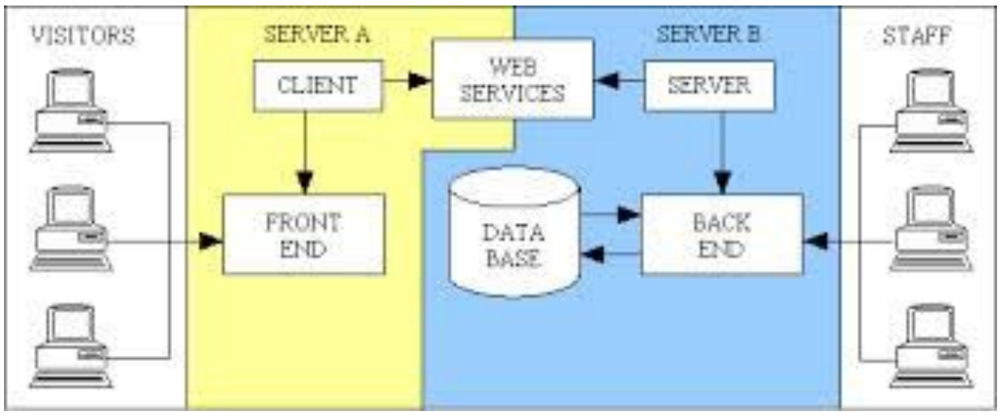
Sauver l'option

Configurer le traçage

URL http://site.com/ | Charger site | AJOUTER événement

URL Hostname

Fig. 3 - Front-end and back-end development



Appendix A - Table of weekly tasks

Week	Tasks	Completed weekly material
1	Discover ClarolineConnect and Trace-Me extension, Kango Framework, JSON etc. ; discuss and suggest a solution for the Trace-Me extension	finished translation of the option page of Trace-Me
2	Translation of the pop-up window and notifications of Trace-Me	update of the progress on the Trace-Me plugin
3	discover AJAX, jQuery and JSON	
4	Translate the Trace-View	finished translation of the Trace-Me plug-in
5	Discover and install Wamp; front-end vs. back-end development; research and discover a solution for the Edit and Translate module	
6	work on Edit and Translate module	suggested a completed plan of the architecture of the module
7	work on Edit and Translate module according to plan - front-end development	organization of files and folders, collection of web content
8	continued work on Edit and Translate	improved web design of the module, functionality to edit button
9	continued work on Edit and Translate, troubleshooting,validation and optimization investigation	functionality to save button, troubleshooting
10	report feedback, investigate complications and improvement options	

Appendix B - Translate and Edit module JS

```
//declare tags to edit and translate
    var topcells = document.getElementsByTagName("th");
    var labels = document.getElementsByTagName("label");
    var buttons = document.getElementsByTagName("input");
    var spans = document.getElementsByTagName("span");

//load new screen text if existent
     $(document).ajaxComplete(function(){
        if(localStorage["mydata"] !== undefined)
        {
          localdata = JSON.parse(localStorage["mydata"]);
          changeLang();
        }
      });

      $.ajax({  //load the JSON database
                type: "GET",
                url: "./Module-translation/library.json",
                crossDomain: true,
                dataType: 'json',
                success: function(data) { libData = data;  }
                  });
//call method on button clicked
     $('#edit-btn').on("click", function() {
          readContents();
     });
//change language on flag click
        $('#EN').on("click", function() {  lang = "EN";
         changeLang();
          });

function changeLang()
      {   //for every found tag search and replace text on screen
                 for(i=0;i<topcells.length;++i)
                {
                  attrib = topcells[i];
                  searchAndReplace();
                }
                for(i=0;i<labels.length;++i)
                {
                  attrib = labels[i];
                  searchAndReplace();
                }
                          (***)
              }

function searchAndReplace()
{
    $(libData).each(function(i){

                         scrVal = attrib.innerText;
                         libValue = libData[i].oldValue;
                                     (***)

      if(lang !== undefined) //if language is changed
      {
          if(lang == ”EN”) { newValue = libData[i].textEN; }

                if(scrVal.indexOf(libValue))
//if the screen text is the same as a string in the library,replace it with its translation
                      {
                        attrib.innerText = newValue;
                      }
       }
     });

     if(localStorage[”mydata”] !== undefined) {
//if text has been modified, replace screen text with the modified text

             $(localdata).each(function(i){

                 oldinput = localdata[i].oldValue;

                 if(oldinput.indexOf(scrVal)>-1 && scrVal !== ””)
                         {
                          newValue = localdata[i].inputValue;
                          attrib.innerText = newValue;
                         }
               });
     }
  }

function editMode() {

        $(attrib).mouseover(function() //highlight the section on hover
        {  //skip empty cells
         if ($(this).text() !== ””)
           {   $(this).css(”background-color”,”#FFFFA3”);  }
        });


        $(attrib).on(”click”, function()
          {      //replace the section with a input on click
           scrVal = $(this).text();

             if ( scrVal !== ””)
               {
                    $(this).wrap(”<input type=’text’
                id=’”+scrVal+”’ value=’”+scrVal+”’></input>”);
                }

                     var inp = document.getElementById(scrVal);

                      inp.onchange = function() {  //add values to array
                        oldValues.push(scrVal);
                        inputValues.push(inp.value);
                        $(this).css("border", "2px solid green");
                      }
            });
i++;
}
      }

    function writeTable() {

         for(i=0;i<inputValues.length;i++)
         {   //write changes to an object
           inputValue = inputValues[i];
           oldValue = oldValues[i];

       mydata  = { oldValue: oldValue, inputValue: inputValue};
       myObj.push(mydata);

         }

         if(localStorage["mydata"] !== undefined)
         {  //parse the object if existent
          local = JSON.parse(localStorage["mydata"]);
          for (var i = local.length - 1; i >= 0; i--) {
              myObj.push(local[i]); }
         }

         localStorage["mydata"]=JSON.stringify(myObj);
         window.location.reload();
```

Sources

jQuery. 2014. *jQuery API Documentation*. [Online]. [Last Accessed 10th June 2014]. Available from: http://api.jquery.com/

KangoExtensions. 2014. *Kango Framework Documentation.* [Online]. [Last Accessed 10th June 2014]. Available from: http://kangoextensions.com/docs/index.html

LIRIS/ CNRS. 2014. *Team SILEX : Supporting Interaction and Learning by Experience.* [Online]. [Last Accessed 10th June 2014]. Available from: http://liris.cnrs.fr/equipes?id=44

W3C. 2013. *Accessibility.* [Online]. [Last Accessed 10th June 2014]. Available from: http://www.w3.org/standards/webdesign/accessibility

W3schools. 2013. *AJAX Tutorial.* [Online]. [Last Accessed 10th June 2014]. Available from: http://www.w3schools.com/ajax/default.ASP