Izzat Nadiri

WEB-DESIGN WITH JOOMLA CMS

Extension development using jQuery library

Bachelor's Thesis
Information Technology

February 2013

MIKKELIN AMMATTIKORKEAKOULU
Mikkeli University of Applied Sciences

MIKKELIN AMMATTIKORKEAKOULU
Mikkeli University of Applied Sciences

Date of bachelor's thesis
June 17, 2012

Author ©
Izzat Nadiri

Degree program and option
Information technology
Programming

Name of bachelor's thesis
Web design with Joomla CMS: extension development using jQuery library

Abstract:
MHG Systems is worldwide known ERP company. Starting from 2011 MHG Systems started upgrading website. My task was to create a website with modern modules for them. The goal was to create a modern website, which should be attractive and eye catching. The target group was business companies not only from Europe, but also from Russia, China and other countries.
The aim was to create a website with alluring moving plugins. As a base, jQuery library was chosen. jQuery worked perfect for this project, since it is very light and easy to implement.
Several modules were developed. Language switcher, drop down menu, slide-up menu, slide-in box are one of those. All these modules are important and popular nowadays.
Language switcher allows us to change the language of website with nice drop down menu which looks very neat and attractive. It works smooth and doesn not take too long to load.
Slide-in box is a module that pops in a box with information and links after scrolling the article to the certain distance.
Slide-up menu is a speed navigation menu, which has heading texts. Those texts slide down and a picture appears instead when they are hovered over. Those pictures have hyperlink, and they would lead us to required url.
Drop down menu does what it says. However, it does not use jQuery. The reason it is included into this project, is for contrast. The idea is to show the difference between module customized with jQuery and the module customized without it.
Last issue was a conversion of jQuery modules in Joomla CMS compatible extensions. This allowed us to continue using Joomla in our project and combine jQuery with CMS.
As a result we got 4 amazing and light modules which made the website look nice and work fast without any lags. These modules clearly show the advantages of jQuery over other libraries and methods of customizing. jQuery can be utilized free of charge, and any questions will be answered in its forum which is active 24/7.

Subject headings (keywords)
jQuery, Joomla, CMS, Web-Development, JavaScript, CSS, PHP, Plugin, Module, Extension.

Pages	**Languages**	**URN**
80	English	

Remarks, notes on appendices

Tutor	**Employer of the bachelor's thesis**
Timo Mynttinen	MHG Systems Oy

CONTENTS

1 INTRODUCTION

Websites are inevitable parts of information technology. Using Internet we are automatically using websites. Almost every company, no matter if it is small, medium or big, has its own website. There are different types of websites:

1. Mobile Device Websites
2. E-commerce Websites
3. Online Business Brochure/Catalog
4. Personal Websites
5. Photo Sharing Websites
6. Informational Websites
7. Blogs
8. Etc.

Each type of website has its own unique functions, outlook, content, modules etc.

Nowadays, webs design is very popular, as it is so easy to master and use in a business. Couple of decades ago, doing website was just writing down thousands of lines of code. It would require us to write a code for every detail. However, it is not as complicated today. Nowadays we have CMS (Content Management System) which is an amazing technique of doing websites faster and easier.

One of the most popular CMS nowadays is Joomla. Joomla has several advantages, such as it is very easy to use, it is open source, it has many tutorials etc. Moreover, users who can create their own modules can sell them in Joomla website.

Substantially, every CMS, including Joomla, already has its own ready functions that we just implement easily. For example, if we want to give a website a nice look, we can choose one template for it. However, even if it is so simple to implement, professional websites should not use open source templates or modules. Professional websites should have their own unique outlook and functionalities.

One of the main parts of Joomla is its modules (extensions). If we want some new functionality in our website we just have to go to joomla.org and download an extension. For example, if you want to have newsletter on a website you just have to download and install it a newsletter module.

Yet again, for professional websites it is better to have own unique extensions rather than download existing ones. It is possible to develop your own modules for Joomla. Development of Joomla modules was the main topic of this project.

We can use variety of programming languages to customize Joomla module. We can use PHP, JavaScript etc. However, in my project work I am going to describe how to customize modules using JavaScript library called jQuery.

jQuery is extremely popular nowadays. Even Google has some modules which use jQuery. For example, when we type text in search box, we can see the results smoothly changing while we type. These are effects of jQuery library.

Nowadays, we can see new features on websites, such as pop ups, slide ins, slide outs, accordions, image rotators, drop downs etc. All these features give websites nicer look and easier usage. Almost all of these features are achieved using jQuery.

jQuery does not require long time to load and get ready for usage. For example, if we compare it to flash, jQuery does not conflict with other modules on pages unlike flash.

In this project work I am going to discuss and describe modules I have customized for Joomla using jQuery.

First module is called slide in panel. This module is very handy and very popular nowadays. I hoped to download it from Joomla website, but was surprised that Joomla did not have it yet. Consequently, I had to customize it myself. The basic idea of that slide in panel is that, when you read an article and you scroll down, at some point a box comes in from lower right corner with some text and link, recommending reading another article.

Next module is called image rotator. This module is meant for using as a header. I customized this module because almost all other Joomla's image rotators are based on flash, which is not flexible. However, jQuery image rotator is very flexible and easy to use. The idea of image rotator is changing images one after another on main page, as a slideshow.

Speed navi is a module that makes navigation faster and nicer. It is basic module with that is located on main page and is used for accessing some important site links faster. Also I used using jQuery to make it look professional. It changes when mouse is hovered on it.

Module language switcher is very nice drop down module that allows us to change website language easily. It looks very nice and uses jQuery to give it drop down effect.

2 SPECIFIC AIMS

2.1 General Purpose

What is CMS?

A content management system (CMS) is a web based application that helps users to create, edit, delete and organize content on a website. Essentially, using CMS is very simple and does not require any prior education in the field of web design. However, knowledge in programming allows certain advanced users to improve and widen functionalities of CMSs. (Rahmel, 2009, 3)

There are hundreds of content management systems available nowadays. CMSs are both closed source and open source. Closed source CMSs are CushyCMS, Unify, Sitecore etc. Whereas, open source CMSs are Joomla, Drupal, Mambo, Nakid etc. (Rahmel, 2009, 4)

Figure-1. CMS brands

CMS usually consists of two sides:

- Content Management Application (CMA)
- Content Delivery Application (CDA)

CMA, as its name implies, allows users to manage data and information. This side of CMS lets us create, delete, edit, copy, duplicate and do other commands with files or data for the website on server. (Bellamy, 2011, 11)

CDA, essentially puts all this information that users modify or create using CMA to the website, making it viewable by other users. (Bellamy, 2011, 12)

Using CMS we usually work with aspects such as:

- Templates

- Categories
- Articles
- Extensions (Modules)
- WYSIWYG
- etc.

2.2 Benefits of CMS

Every one of us has seen an advertisement of internet saying something like: “Create a website in 10 minutes!” or “Get your own webpage today!” Those ads are very attractive, but do they actually work as they say? There are many free website providers such as Weebly, Webs and they operate quite well. However, they have many limitations, such as templates, modules, copyright etc.

Using such providers, we will always have to stick to their ads, and in case we want to get rid of them we will have to pay.

What makes CMS a better choice?

Using CMS we have an access to every bit of data. We can change anything we want. It depends on our knowledge, if we can do that or not. CMS mainly provides us a GUI and allows us manually add, remove and modify articles and plugins. These options are very important for beginners who do not want to spend a reasonable amount of money on their website. CMS are usually free and customizing our own modules we can save a lot of money! (Bellamy, 2011, 15)

Consequently, CMS is very beneficial tool that helps us bypass hard process of web designing.

2.3 Basic components

Templates

As Rehmel mentioned templates are used to give a nice look to a front end of website. Templates are usually very simple and easy to customize. To customize basic CMS template, CSS, HTML and PHP knowledge are required. However, if user lacks that knowledge he can download free templates or buy already customized and ready templates online.

Articles

Article and categories are aspects related to the website content. CMS usually have graphical user interface that allows users easily to write and edit content of website.

Extensions

Extensions are very important part of every CMS. Extensions are also referred to as "modules". Essentially extension is small application that runs on website.
(Rehmel, 2009, 241)

Figure-2. Joomla extentions

One of the main components of any CMS is WYSIWYG. It stands for "What you see is what you get". WYSIWYG mainly is a text editor used in CMSs. It is a very useful tool. More advanced options WYSIWYG has, more professional the website will look. (Rehmel, 2009, 243)

WYSIWYG editors usually have functions such as:

- embed flash
- insert picture
- embed video
- change font-style, font-color, font-size
- embed HTML,PHP, JavaScript codes
- etc.

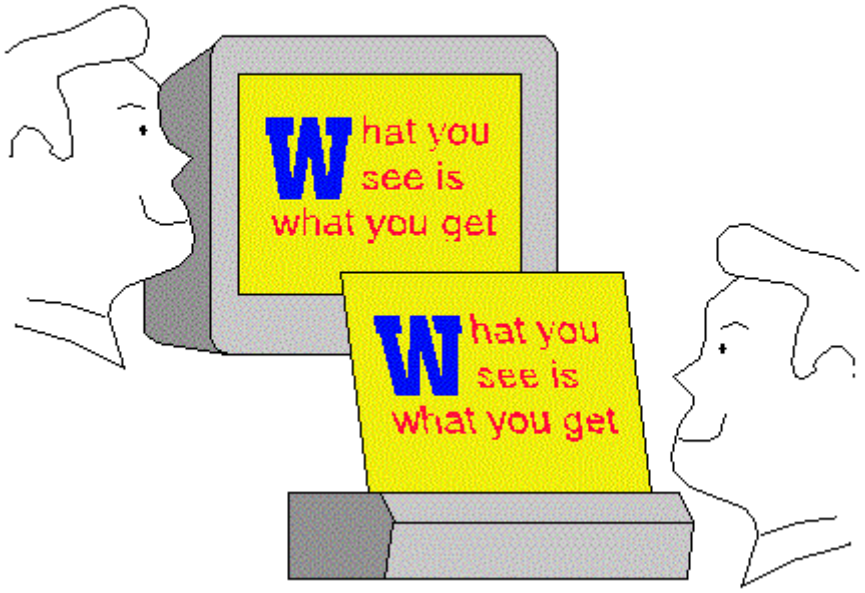


Figure-3. WYSIWYG

Joomla has several popular WYSIWYG editors such as JCE,CKE editor, JCK editor, TinyMCE etc. (Rehmel, 2009, 280)

The actual aim of CMS is to provide user database for a website that makes it easier to control and identify users their access to the website and their roles. Using CMS we can assign certain roles to users that will prevent them from doing things that they should not do with website content. For example, being admin user on website allows full control of content, while a simple user can just utilize the system and cannot modify any content. (Bellamy, 2011, 225)

2.4 Onset

Installation of CMS

Learning to use Content Management Systems is not the final step of web design. There are many other steps that we have to do before getting to actual web design. One of those steps is installation of CMS.

Many people think that to have a website for practicing their skills they have to pay monthly fee for using web space. However, they are wrong, because it is possible to run website on local computer, utilizing it as a server. Essentially, it is possible to install any CMS on localhost and use it as actually website for practicing.

Required details

Web Server Softwares

To install CMS on both, local and remotely web server software pack is needed. There are many free and open source web server software programs such as MAMP, WampServer, AMPPS, EasyPHP etc. However, the most popular web server software is XAMPP. It is widely use nowadays, mainly because it is powerful, free and open source. XAMPP is mainly a combination of Apache, PHP and MySQL.

Usually XAMPP is installed to "C" drive into folder "xampp". In folder "xampp" there is another folder, called "htdocs". This folder is actually our local web space. To install or transfer some applications to our server we just dump them in that folder.

To install CMS application to local computer, download installation pack. It usually comes in .zip format. Unzip it to the "htdocs" folder, and then go to any web browser and in address line type "localhost/NameOfInstallationPackage". Next step is to install and configure CMS properly, set up database, create username with password etc. (Rehmel, 2009, 65-70)

Figure-4. XAMPP

File Transfer Protocol software (FTP)

After installation of CMS on our web space, we need to upload our data. Easiest way of doing it is via FTP software. The most well-known FTP software is FileZilla. However, there is a long list of other open and closed source FTP softwares:

1. EFT Server
2. NASLite
3. Robo-FTP
4. Titan FTP
5. War FTP
6. etc.

Main point of FTP server software is that it allows us direct file transfer to our web space. It requires some information to connect to the server, such as, password and username. After the connection is built, we can freely upload data, such as pictures, videos, music and other data to our server. (Rehmel, 2009, 325)

Figure-5. FileZilla

2.5 Major objectives

Joomla module creation

This section of the thesis is the most essential, because the point of this project is based more on Module Development rather than on other issues.

As Joomla CMS is open source, many users develop their own modules for it. Some of them upload them for free usage others apply copyright and ask for money or donations. Joomla 1.0 was first in 2005, and during these years thousands of modules were developed for users. However, nowadays we can need some modules that we cannot find. This is when we need develop our own Joomla module. (Rehmel, 2009, 246)

Nowadays websites have changed a lot. Comparing to websites of 10 years ago today's websites look fancier and more professional. Most of the large-scale websites have moving, sliding and fading modules in it. It is known as a website fashion of 2010's. (Bellamy, 2011, 291)

Simonelly claims, that there are several ways of applying those effects (moving, sliding and fading) to a website, such as Flash, HTML 5, jQuery etc.

Figure-6. jQuery

However, it is also known that methods such as Flash are quite heavy and take a lot of time to browse and buffer. Would we like the website that takes 3-4 minutes to load? I would say no!

However, this problem was overcome by mean of jQuery. Using jQuery, we can achieve the same effects without any extra load on our server. jQuery effects work much faster and easier on web space or local machine.

As I mentioned earlier, jQuery is a library of JavaScript that includes many nice effects. Nowadays we can find many modules with jQuery in Joomla website, and as we can see there, they are quite popular and users are completely satisfied with them.

Creating a jQuery module is just one step of the issue. Next is to convert the module into Joomla extension, so that Joomla CMS will be able to recognize it on the server and utilize it as its own extension.

2.6 Expected outcome

As an outcome we tend to get a nice, eye-catching plugin that makes our website look professional and complete. jQuery is very easy to learn and use, unlike other programming tools and languages. Using jQuery we can easily utilize properties such as CSS, PHP and HTML.

All of us have seen websites of great corporations such as, Apple, Blackberry, IBM, Nike etc. Their web pages look very attractive and alluring. However, sometimes we also face some problems loading that kind of overloaded websites. Great corporations usually use Flash as a base of their sites. Although jQuery can give the same tempting outcome, it is not going to lag or buffer as websites made on Flash. (Bibeault, Katz, 2010, 235)

jQuery is a tool that can make a simple website look alluring and attractive. As an outcome we can use the plugin wherever we want, not only in CMS. We can use it on any kind of website. Main goal here is to customize a plugin and come up with needed lines of code.

3 BACKGROUND AND SIGNIFICANCE

3.1 How to use jQuery?

To start with plugin (module) customization, we have to get familiar with jQuery first. Later on we will learn how to download, install, test and use jQuery library on our local machine.

First of all we need to install software called XAMPP to be able to view the code or plugins that we customize. We can download it from the link below:

http://www.apachefriends.org/en/xampp.html

After installation we will get a folder in /C drive called “xampp”. In that folder we can find another folder called “htdocs”. This folder is our local “webspace”. Whatever we put in that folder we can view in browser typing in

http://localhost/*NAMEOFTHEFILE*.

Next we need to create a folder and call it whatever we want for ease of use. Inside that folder we create a PHP file and call it “**index.php**” (Alex Garrett, Introduction to jQuery, 2011)

3.2 Where to get jQuery?

First of all, jQuery is an open source, and is available absolutely free. To download jQuery library, we need to go to the link www.jquery.com. There we will see the section "Download (jQuery)". However, there are two options in that section. First option is PRODUCTION (32KB, Minified and Gzipped) and second is DEVELOPMENT (252KB, Uncompressed Code).

Obviously there is some difference between those packages. Mainly, first package, which is smaller in size, is intended for simple users without any professional and development skills. Second package is for developers who want to access broader uncompressed code which gives them more options for development. (Alex Garrett, Introduction to jQuery, 2011)

Figure-7. jQuery download guideline

Installation

After we download production version, a page with some code opens up in the new window of browser. We should go ahead and copy that code to our clipboard. Later on we should create a blank text document and insert that jQuery code in there. After that we change its extension to ".js "and call it jquery.js. Then we should put that file into folder that we create and call js. "js" folder should be located in the same folder with the file "**index.php**" Consequently, now we have a folder "js" with a file called "**jquery.js**" that contains the code we downloaded from jQuery website.

Now is a right time to utilize our "**index.php**". We need to create a connection between jQuery library that we downloaded and "**index.php**". To do that we open up

our "**index.php**" file and modify it. We are now going to link "**index.php**" file to "**jquery.js**" file. . (Alex Garrett, Implementing jQuery, 2011)

To do that we just type the line:

```
<script type=”text/javascript src”ja/jquery.js”></script>
```

```
1   <!doctype html>
2   <html lang="en">
3   <head>
4   <meta charset="utf-8">
5   <title>jQuery Example</title>
6   </head>
7   <body>
8
9   <script type="text/javascript" src="js/jquery.js"></script>
10  </body>
11  </html>
```

Figure-8. Source code 1

3.3 Testing jQuery

Garret claims that if we open "**index.php**" in browser and it works, it does not mean that our jQuery code works fine. The reason for that is that jQuery is just a script.

To test our jQuery we need to write some very simple code that will hide a paragraph. To do that we need to type the code below into our "**index.php**" file's body part:

```
<p onclick=”$(this).hide();”>This text will be hidden if
you click it!</p>
```

```
1   <!doctype html>
2   <html lang="en">
3   <head>
4   <meta charset="utf-8">
5   <title>jQuery Example</title>
6   </head>
7   <body>
8   <p onclick="$(this).fadeOut();">This text will be hidden if you click it!</p>
9   <script type="text/javascript" src="js/jquery.js"></script>
10  </body>
11  </html>
```

Figure-9. Source code 2

3.4 Learning jQuery

”Ready” function

Basic purpose of jQuery is eye-catching movements. jQuery modules are very noticeable and attractive by the way of their activation or appearing. However, sometimes we need to take a certain action for module to start. As we see on the **Figure-9** , function onclick activates the module when we click on a certain item, which in our case is text itself.

However, sometimes we do not want users to click anything on the web page, but items appear themselves at certain time. “Ready” function is one of the useful tools we need to know in jQuery.

To use “Ready” function, we first of all need to come up with a certain item, which we want to pop up or appear. For example, we can use a text. The idea of “Ready” function is that it allows module to appear right after the page is completely loaded.

It means when we open our webpage, we will not see the text, until all the other items are loaded and ready for user to view. (Chaffer, Swedberg, 2009, 391)

Now let’s take a look at “Ready” function in practice.

Yet again we will continue from where we left off in previous “Testing” example. First of all we need to create a new file called “ready.js” in the folder “js”. Then we should include that file in our initial “index.php” file.

Now “index.php” file should look like this:

```
1   <!doctype html>
2   <html lang="en">
3   <head>
4   <meta charset="utf-8">
5   <title>jQuery Example</title>
6   </head>
7   <body>
8   <div id="message" style="display:none;">Welcome to my website</div>
9   <script type="text/javascript" src="js/jquery.js"></script>
10  <script type="text/javascript" src="js/ready.js"></script>
11  </body>
12  </html>
```

Figure-10. Source code 3

Line 10 in that picture represents how we included file “ready.js” into our main “index.php” file.

Now let’s modify “ready.js” file and write a code that utilizes “Ready” function. Here is how our “ready.js” file should look like:

```
1  $(document).ready(function(){
2  $('#message').fadeIn('slow');
3  });
```

Figure-11. Source code 4

On **Figure-11** we see the file "ready.js". These lines of code mean that when the document is ready (fully loaded), item with "id=message", which is our text in "index.php" file should fade in. The speed of the motion is set to slow. However, it is also possible to set it fast, or just numerically type in the speed of motion.

"Load" event handler

In previous example we reviewed the situation when our item appears when whole page is loaded. What if we need to see our item when the certain element is loaded, not the whole page? jQuery has a solution for that! We should use "load" even handler. (Bibeault,Katz, 394)

Let's take a look at a situation when we have a large image on our web page. When the image is loaded we want a text box to pop up saying that our picture is successfully loaded. "Ready" function will not do that, because we need one item to be loaded. Using "Load" event handler, we can perform that task easily. We just write a code that tells, to pop up a text box only, when the picture is fully loaded. Let's see that on practice.

First let's create a new file called "load.js" in js folder. Then we should modify our "index.php" and include that newly created in there:

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf-8">
5  <title>jQuery Example</title>
6  </head>
7  <body>
8  <img id="image" src="helsinki.jpg"/>
9  <script type="text/javascript" src="js/jquery.js"></script>
10 <script type="text/javascript" src="js/load.js"></script>
11 </body>
12 </html>
```

Figure-12. Source code 5

On line 10 we see the code that includes"load.js" file in our initial "index.php" file.

On line 8 we see the code that adds an image to our site. The image has an id="image".

Now we are done with file “index.php”, so let’s modify file “load.js”.

```
1  $('#image').load(function(){
2  alert('Image is successfully loaded');
3  });
```

Figure-13. Source code 6

On the **Figure-13** we see the code that loads an alert “Image is successfully loaded” only when item with id=”image”, which in our case is our image, is loaded.

Selectors

As Chaffer and Swedberg imply, Selectors are essential part of jQuery library. There are many types of selectors in jQuery library that are very important to know. Essentially, selectors are used to match and select elements from a page and perform an operation over them. Some types of selectors are listed below:

- All selector
- ID selector
- Element selector
- Submit selector
- Text selector
- Multiple selector
- This selector
- Even/odd selector
- Attribute selector
- Contains selector

“All” selector

“All” selector is very a useful tool in jQuery. We can always face a situation when we need to select all the elements on our webpage and perform a task over them. For example, we need to count how many elements we have on our web page. Or sometimes we need to count how many images or music files we have. That is when “All” selector is useful. (Sharp, Burns, 2010, 46)

“ID” selector

Sharp and Burns state, that “ID” selector is not only used in jQuery but also on other cases. “ID” selector is very easy to use and is extremely important. “ID” selector allows us to select some certain elements based on their IDs. First of all, when we use

this tool, we need to assign an ID to an item. For example let's assign and ID to a "div":

```
<div id=aurinko></div>
```

We can attach ID to any item on our page and this makes it easy to manage them.

Later on when we need to modify them in CSS or jQuery we just use their ID and add # to them. For example, if our ID is "aurtinko", then we write it as "#aurinko". (Alex Garrett, "ID" selector, 2011)

"Element" selector

Sometimes we need to apply a certain element, not to every single element on the page. This is when we use "Element" selector. This is also very useful tool. It keeps us from adding ID to every item that has the same element type.

For instance, if we have a page with 5 paragraphs, 3 pictures and a couple of other items and we need to select only paragraphs and perform a task over them. We can do that using Element selector. Paragraph symbol is "p", so in jQuery we just type "p" which indicates that our code is related to all paragraphs on page. (Alex Garrett, "Element" selector, 2011)

"Submit & Text" selectors

Sharp and Burns define that these two types of selectors are very basic and are mainly an example of "Element selector". To select a submit button, we just write its type in a code as in the previous example about "p".

"Text" selector is about selecting a text box. For example, when we want something to happen when we hoover our mouse to the text box we use text as a selector type. (Alex Garrett, Submit selector, 2011)

"Multiple" selector

"Multiple" selector is a combination of selectors. Sometimes we have so many elements that we can group them. For example, buttons, paragraphs, items with IDs etc.

In this case we can select type button, type paragraph and then items with certain ID. (Alex Garrett, "Multiple" selector, 2011)

"This" selector

This selector is very useful and is widely used in many programming languages. "This" selector allows us to drop the name of an item or element if we have already mentioned it earlier. In this case we just type "this" instead. (Bibeault, Katz, 2008, 20)

"Even/odd" selector

Bibeault and Katz mention that “even/odd” selectors allow us to select either even or odd item from certain list. For instance, if we have a table we can select odd or even rows or columns from it. (Alex Garrett, “Even/odd” selector, 2011)

“Attribute” selector

“Attribute” selector is very important. We use “Attribute” selector when we need to select an attribute of an element and not the element itself. For example, we use it when we need to change a title of a button, but not the button itself. (Chaffer, Swedberg, 2009, 24)

“Contains” selector

“Contains” selector is very widely used. We use it when we need to perform an action over some item that user inputs. First of all, we need to make sure that the item the user inputs is already on our page. For example, in all browsers, when we press “CTRL+F”, the small search panel opens up. Then when we type in a text, it highlights it on the page. This is what Contains selector does. (Chaffer, Swedberg, 2009, 29)

Now we will take a look at a practical example of “Contains” selector.

First of all we need to create a CSS file. We need to create a separate folder called “css”. Then let’s create a file called “style.css” in that folder.

Next we need to modify “index.php” to connect it to css file.

Modifying “index.php”, we also need to create a small table out of 4-5 names. Also we need to add a text box where we can type letters.

Then we need to create a file called “contains.js” in “js” folder and connect it to our “index.php” file.

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf=8">
5  <title>jQuery Example</title>
6  <link rel="stylesheet" type="text/css" href="css/style.css">
7  </head>
8  <body>
9  <p><input id="name_search" type="text"/></p>
10 <ul id="names">
11 <li>Matti Suominen</li>
12 <li>Leena Keinänen</li>
13 <li>Petri Saarinen</li>
14 <li>Niina Nurminen</li>
15 </ul>
16 <script type="text/javascript" src="js/jquery.js"></script>
17 <script type="text/javascript" src="js/contains.js"></script>
18 </body>
19 </html>
```

Figure-14. Source code 7

Next step is to modify that contains.js folder and add some code there.

```
$(document).ready(function() {
$('#name_search').keyup(function(){
name_search = $(this).val();
$("#names li:contains('" + name_search + "')").addClass('highlight');

});

});
```

Figure-14. Source code 8

Last step is to modify CSS file to create a new option for plugin.

```
.highlight{
background-color:yellow;
}
```

Figure-15. Source code 9

As a result we get a plugin that works like this:

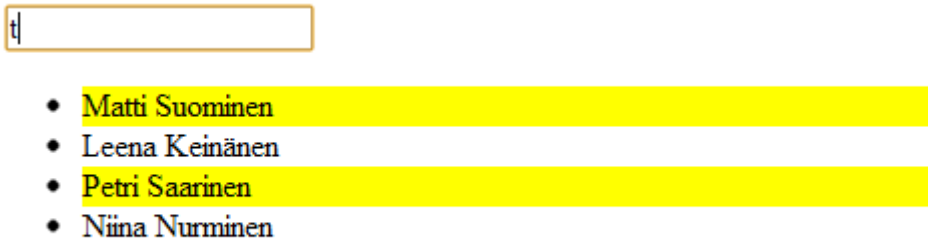


Figure-16. Example image 1

This plugin needs some more modification to work perfectly as a search box of browsers, but it reflects the main idea of Contain selector. (Alex Garrett, “Contains” selector, 2011)

Font size switcher

Sometimes articles in some websites have really small fonts. Website developers should also think about elderly people that find it hard to read small fonts. However, most of sites have font readable only by younger people. There is a way out of this

situation by pressing “CTLR&+”. However, using jQuery we can create a button that allows us to do so.

This topic is also related to selectors. We can call this section a font size selector. The reason for that is that we essentially select font size and perform an actions over it.

First of all, we need to create a js file where we are going to write our code. We will call it “fontswitcher.js”. Then we will modify our “index.php” and connect it to our new, “fontswitcher.js”.

Now let’s add 2-3 paragraphs to “index.php”. Then let’s add links that will make size smaller and bigger.

```
1
2
3
4   ">
5   le</title>
6   t" type="text/css" href="css/style.css">
7
8
9   " id="smaller">Smaller</a> <a href="#" id="bigger">Bigger</a>
10  n pääkaupunki.</p>
11  ital of Finland.</p>
12
13  javascript" src="js/jquery.js"></script>
14  javascript" src="js/fontswitcher.js"></script>
15
16
```

Figure-16. Source code 10

Last step is to write code into “fontswitcher.js” that will make the plugin work.

```
1   function change_size(element, size){
2   var current=parseInt(element.css('font-size'));
3   if (size=='smaller'){
4   var new_size = current-2;
5   }
6   else if (size=='bigger'){
7   var new_size=current+2;
8   }
9   element.css('font-size', new_size +'px' );
10  }
11  $('#smaller').click(function(){
12  change_size($('p'),'smaller');
13  });
14  $('#bigger').click(function(){
15  change_size($('p'),'bigger');
16  });
```

Figure-17. Source code 11

Fundamentally, in this code we measure the font of the text, “parse” it to integer, add or subtract 2 from it and change the font of the text to final result.

Here is what we get as a result plugin:

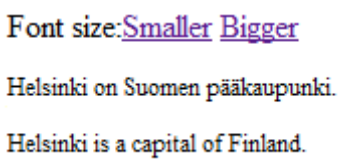


Figure-18.Example image 2

After we press “Bigger” button 4 times here is what we get:

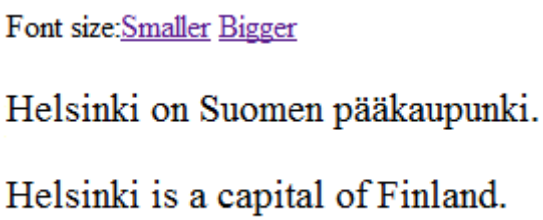


Figure-19.Example image 3

Even though, this small plugin is not so common on websites nowadays, it is a very useful tool that works very fast and easily in any web browser. (Alex Garrett, Font size switcher, 2011)

Toggle

Toggle is widely used tool in any programming language. jQuery library also has this function, that might be very useful in customizing plugins and modules. Toggle mainly means, changing between two options. For example, when we turn the light on and off, we are just toggling it. The same idea stands behind the programming toggle.

Let’s now create a small plugin that will toggle the text that we have in our website. First of all, we need to create a “js” file called “toggle.js”. Then we need to modify “index.php” file and connect it to our “js” file. Next step is to modify “index.php” file and add a button that will allow us to do toggling.

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf=8">
5  <title>jQuery Example</title>
6  <link rel="stylesheet" type="text/css" href="css/style.css">
7  </head>
8  <body>
9  <a href="#" id="click">Click this button to toggle</a>
10 <div id="click_event"></div>
11 <script type="text/javascript" src="js/jquery.js"></script>
12 <script type="text/javascript" src="js/toggle.js"></script>
13 </body>
14 </html>
```

Figure-20.Source code 12

Line number 9 is a “href” which will later be our “toggle button”. Next step is to modify “toggle.js” file and write down the code that will toggle the text.

```
1  $('#click').toggle(function(){
2  $('#click_event').html('Yes');
3  }, function() {
4  $('#click_event').html('No');
5  });
```

Figure-21.Source code 13

Here we see the code that reflects toggle tool. When we press item with id=click, which is our “href”, we toggle “div”, which has an id=click_event, and it changes the text to Yes then No. Here is how it looks like:

Click this button to toggle

Figure-21.Non-pressed

Click this button to toggle
Yes

Figure-22.Pressed once

Click this button to toggle
No

Figure-23.Pressed twice

(Alex Garrett, Toggle, 2011)

Hover

One of the key aspects of jQuery is “hover” function. “Hover” can be used in many cases. For example, when we make a drop-down menu for our website, we want sub-links fall-open when we hover over them, we use “hover” function of jQuery. (Bibeault, Katz, 2008, 121)

In this example we will take a look at a small plugin which shows us the description of the button when we hover over it.

First of all, we create a new “js” file called “hover.js”. Then we modify “index.php” file and connect it to our new “js” file.

Now it is time to modify our “index.php” file.

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf=8">
5  <title>jQuery Example</title>
6  <link rel="stylesheet" type="text/css" href="css/style.css">
7  </head>
8  <body>
9  <a href="#" id="mikkeli">Mikkeli</a>
10 <a href="#" id="helsinki">Helsinki</a>
11 <a href="#" id="turku">Turku</a>
12 <div id="city_feedback"></div>
13 <script type="text/javascript" src="js/jquery.js"></script>
14 <script type="text/javascript" src="js/hover.js"></script>
15 </body>
16 </html>
```

Figure-24.Source code 14

On **Figure-24** we see a code that creates 3 “href”s that have names of three Finnish cities. Then, on line 12, we see a “div” where the description of those cities will appear when we hover on “href”s.

Now let’s edit “hover.js” file and write our code there.

```
1  $('#mikkeli').hover(function(){
2  $('#city_feedback').html('Mikkeli on paras kaupunki');
3  });
4  $('#helsinki').hover(function(){
5  $('#city_feedback').html('Helsinki on suuri kaupunki');
6  });
7  $('#turku').hover(function(){
8  $('#city_feedback').html('Turku on kaunis kaupunki');
9  });
```

Figure-25.Source code 15

Here we see a code that shows us a certain text when we hover over each city.

Here is a result:

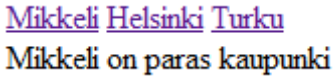


Figure-26.Hover over Mikkeli

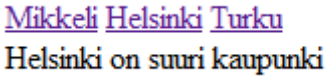


Figure-27.Hover over Helsinki

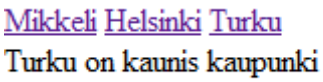


Figure-28.Hover over Turku

(Alex Garrett, Hover, 2011)

Scroll

jQuery has a "scroll" event handler which is very useful. In one of the key modules I have created, the main attribute was scroll. (Bibeault, Katz, 2008, 312)

I have customized a module called slide in box. The basic idea of that module is that when we have a text on our website, for example some article and we scroll down while reading, a small box with ad pops up from one corner. Working technique of this module is that we give some point of the text and id. Later in our "js" file we specify that when our mouse scrolls to the point with that id, a box pops up.

However, now we will overview another module, which just simply shows how "scroll" tool works.

In this module we will create a text box and while we scroll down, we will see on which line we currently are.

To do that, we should first create a "js" file called "scroll.js". Then we should edit "index.php" file and connect it to "js" file.

Now let's modify "index.php" file and add a text box to it and a "div" where the countdown will be shown.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf=8">
<title>jQuery Example</title>
<link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
<textarea id="text" rows=8 cols="20">Suomen tasavalta eli Suomi (r
Suomi kuuluu maapallon lämpövyöhykkeistä pääosin lauhkeaan vyöhykk
<div id="feedback"></div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/scroll.js"></script>
</body>
</html>
```

Figure-30.Source code 16

As we see, we gave an ID to text and “div”.

Now let’s write our code to “js” file.

```
$('#text').scroll(function(){
var scroll_pos = $('#text').scrollTop();
$('#feedback').html('Scroll is on the line '+ scroll_pos);
});
```

Figure-31.Source code 17

Substantially on the **Figure-31** we see the code which has a variable called scroll_pos which equals to the line number of current position of mouse scroll.

Here is a final result:

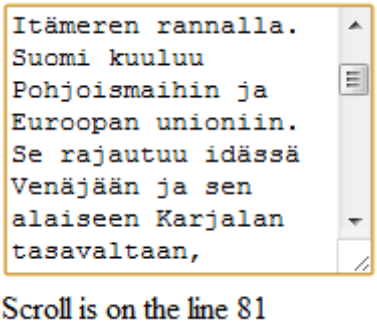


Figure-31.Example image 4

(Alex Garrett, Scroll, 2011)

Character counter

Next small plugin is about a character counting. This tool is very common in social networking such as Facebook, Twitter, VK, MySpace etc. Using this plugin we can control how long is, for example, the comment on a picture or a private message.

First of all, to create this module we need to create a “js” file called “counter.js”. Then we modify “index.php” file and connect it to “js” file.

Now let’s modify “index.php” file and add text box and feedback “div” to it.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Example</title>
<link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
<textarea id="text" rows=8 cols="20" maxlength="60"></textarea>
<div id="feedback"></div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/counter.js"></script>
</body>
</html>
```

Figure-33.Source code 18

Now let’s modify “counter.js” and write down our code.

```
$(document).ready(function(){
var text_max=55;
$('#feedback').html(text_max+' characters remaining');
$('#text').keyup(function(){
var text_length=$('#text').val().length;
var text_remaining=text_max-text_length;
$('#feedback').html(text_remaining+' characters remaining');
});
});
```

Figure-34.Source code 19

In this code we mainly create a variable which is text_max and equals 55. Then we create a variable text_length which is a length of the text we type. When we create a variable called text_remaining which equals the difference between text we type and maximum allowed size of text.

Here is what we get:

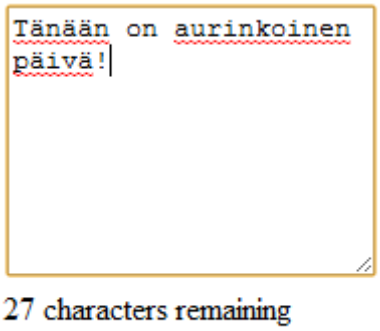


Figure-34.Example image 5

(Alex Garrett, Character counter, 2011)

Val (Value)

"Val" is a very important and useful attribute in jQuery. In previous example, we used it, and the whole example was based on "val" attribute. Essentially, "val" counts the value of an item.

For example, when we have a text field, it can count the number of letters in the text field.

In our next example we will see how it works.

First of all, we need to create a "js" file and call it "val.js". Then we need to modify "index.php" and connect it to our "js" file.

Next we need to edit "index.php" as below:

```
1   <!doctype html>
2   <html lang="en">
3   <head>
4   <meta charset="utf=8">
5   <title>jQuery Example</title>
6   <link rel="stylesheet" type="text/css" href="css/style.css">
7   </head>
8   <body>
9   <input type="text" id="name"/>
10  <div id="area"></div>
11  <script type="text/javascript" src="js/jquery.js"></script>
12  <script type="text/javascript" src="js/val.js"></script>
13  </body>
14  </html>
```

Figure-35.Source code 20

Here we essentially create a textbox and give it an ID. Then we create a "div" to show the count in it later.

Now it is time to write a code to our "val.js" file.

```
$('#name').keyup(function(){
var name=$('#name').val().length;
$('#area').text(name);
});
```

Figure-36.Source code 21

The code we wrote, works so, that when we type in something it takes out of it its value and displays it in “area div”.

Here is what we get:

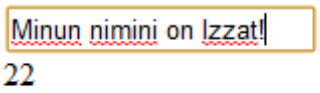


Figure-37.Example image 6

(Alex Garrett, Val, 2011)

Fade in

One the main tools in this thesis work is “fade in” event handler. It does exactly what it says, it fades objects in.

In the example we will see how an image will fade in when the page is ready and loaded.

First of all, as usual, we create a “js” file and call it “fadein.js”.

Then we modify “index.php” file and connect to our “js” file.

Next step is to modify “index.php” and add an image to it and give it an ID.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf=8">
<title>jQuery Example</title>
<link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
<img id="image" src="lippu.jpg" alt="Lippu" style="display:none;"/>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/fadein.js"></script>
</body>
</html>
```

Figure-38.Source code 22

Then we write a code to “fadein.js” file.

```
$(document).ready(function(){
$('#image').fadeIn(5000);
});
```

Figure-39.Source code 23

This code does what it says, fades image in.

Figure-40.Example image 7

(Alex Garrett, Fade in, 2011)

Fade toggle

Previously we took a look at toggle tool. This tool is very useful and is used broadly on internet. It can be connected to many plugins to make them usable. For example, users will not be happy if they press a button and something pops up, and they will not be able to hide it anymore. This is why we use toggle function.

Next example will combine two great tools together: fade and toggle.

This and previous examples will be very much the same, though the only difference will be the button functioning. In this case button will toggle the image, not just fade in or out.

Now let's get to the example.

First of all we need to create "togglefade.js" file. Then we need to connect it to our "index.php" file.

Let's modify "index.php" file as below:

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf-8">
5  <title>jQuery Tutorial</title>
6  </head>
7  <body>
8  <p><input id="inout" type="button" value="Fade in/out"/></p>
9  <img id="image" src="lippu.jpg" alt="Lippu"/>
10 <script type="text/javascript" src="js/jquery.js"></script>
11 <script type="text/javascript" src="js/togglefade.js"></script>
12 </body>
13 </html>
```

Figure-41.Source code 23

Here we just added a picture and gave it an ID. Then we created a button and also gave it an ID.

Next let's customize a code in "js" file.

```
1  $('#inout').click(function(){
2  $('#image').fadeToggle();
3
4  });
```

Figure-42.Source code 24

Code above means that when button with an id="inout" is clicked, the picture with an id="image" is toggled. (Alex Garrett, Fade toggle, 2011)

Slide down

Next tool that is very important is slide down. It has the same idea as slide in or slide out. However, the main difference is that it slides from up to down. Usually slide down slides from the top of the page.

On famous webpages such as twitter, 9gags and many others we can see those plugins in function.

Usually these kinds of modules are somehow related to "sign in" function.

All of us probably have seen buttons on websites which drop a "sign in box" after they are pressed.

In the next example we will see a text that will slide down when the page is ready and loaded.

First of all we need to add "css" file to our folder. Let's call it "style.css".

Next let's create a "js" file and call it "slidedown.js".

Now we must connect it to our "index.php" file.

After all these steps we need to modify "index.php" and add some new lines of code to it.

Final result should look like below:

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf-8">
5  <title>jQuery Tutorial</title>
6
7  <link rel="stylesheet" type="text/css" href="css/style.css" />
8  </head>
9  <body>
10
11 <div id="top_message">We can see you're not logged in.<a href="#"> Sign up?</a></div>
12 <script type="text/javascript" src="js/jquery.js"></script>
13 <script type="text/javascript" src="js/slidedown.js"></script>
14 </body>
15 </html>
```

Figure-43.Source code 25

In the **Figure-43** we see how we can connect “css” file to “php” file on the line 7. Then we see the text with id=”top_message”. This is a text that will slide down. Also we need to edit “css” file we created to make the text look nice and hide it to make is slide down later.

```
1  body{
2  margin-top:0px;
3  }
4  #top_message{
5  display:none;
6  margin-left:auto;
7  margin-right:auto;
8  text-align:center;
9  width:700px;
10 background:#000;
11 padding:20px;
12 font-size:20px;
13 color:#fff;
14 }
```

Figure-44.Source code 26

Now let’s write our code to “slidedown.js” file.

```
1  $(document).ready(function(){
2  $('#top_message').slideDown('slow');
3  });
```

Figure-45.Source code 27

After this code, the final result will look like this:

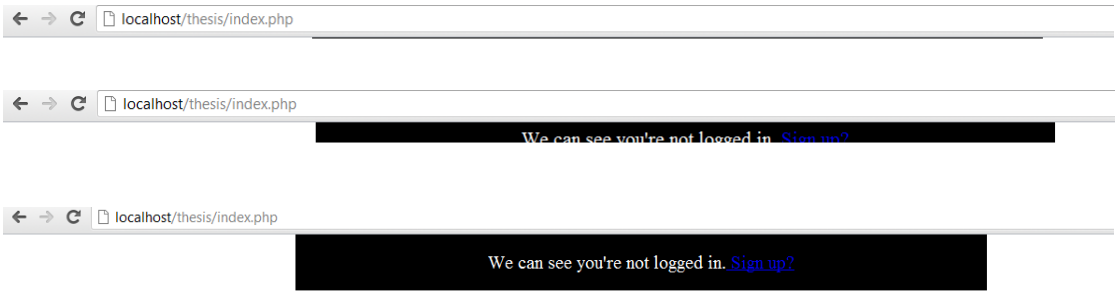


Figure-46.Example image 8

(Alex Garrett, Slide down, 2011)

Stop

"Stop" function is not very common in jQuery because it is waved by "toggle" function. Fundamentally, "stop" button stops some actions from happening.

In our next example we will prevent a picture from sliding by clicking "stop" button.

In this example we will review a plugin that slides the picture and when we press stop button it stops.

First of all, we need to create "js" file called "stop.js". Then we connect it to "index.php" file.

Now let's edit "index.php" file and add buttons and picture.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<p><input id="start" type="button" value="Start"/> <input id="stop" type="button" value="Stop"/></p>
<p id="image"><img src="lippu.jpg" alt="Lippu"/></p>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/stop.js"></script>
</body>
</html>
```

Figure-47.Source code 28

Now let's edit our "js" file and write our code.

```
$('#start').click(function(){
$('#image').slideToggle(5000);
});
$('#stop').click(function(){
$('#image').stop();
});
```

Figure-48.Source code 29

This code mainly, toggles a picture when we press a button with id="start" and stop it when we press button with id="stop".

Here is how it looks like:

Figure-49.Example image 9

Figure- 50.Example image 10

On the first picture we see our page without any buttons pressed. On the second picture we see how the image slides and stops when we press “stop” button. (Alex Garrett, Stop, 2011)

Delay

Delay is very important tool in jQuery which is used usually to start some action after certain time. For example, if we want to shut down our laptop we can use a “delay” function not to shut it down straight away, in case we still need to do some task on it.

In our next example we will take a look at a situation when we press a button and a text fades away, but not right away.

To make this example work, we need to create a “js” file first and call it “delay.js”. Next, we connect it to our “index.php” file.

Then we need to modify “index.php” file and make it look like below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<p><input id="vanish" type="button" value="Vanish"/></p>
<p id="para">I will disappear!</p>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/delay.js"></script>
</body>
</html>
```

Figure-51.Source code 30

Here we just create a button and give it an id=”vanish”, also we create a paragraph that will disappear later, and give it an id=”para”.

Next we modify “delay.js” file and write our code.

```
$('#vanish').click(function(){
$('#para').fadeOut(5000).delay(3000).fadeIn();
});

```

Figure-52.Source code 31

Code above means that when we press a button with id=”vanish”, text with id=”para” disappears after 3 seconds.

Here is how it looks like:

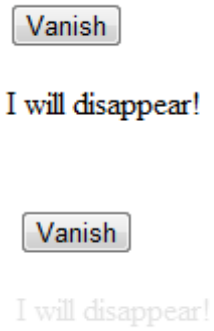


Figure- 53.Example image 11

(Alex Garrett, Delay, 2011)

Gallery fading effect

Gallery fading effect is widely used all over the internet. We all have seen websites where some pictures are dim, and only when we hover on them they attain their 100% brightness and opacity. This effect can usually be seen on picture gallery websites.

In our example, we will create a plugin that will have 2 pictures which are originally dim. Then when we hover on them they become brighter.

First of all, we need to create a "js" file and call it "opacity.js". Then we need to connect it to our "index.php" file. After that we need to modify "index.php" file as below:

```
1  <!doctype html>
2  <html lang="en">
3  <head>
4  <meta charset="utf=8">
5  <title>jQuery Tutorial</title>
6
7  <link rel="stylesheet" type="text/css" href="css/style.css" />
8  </head>
9  <body>
10 <img class="fadeto" src="lippu.jpg"/>
11 <img class="fadeto" src="lippu.jpg"/>
12 <script type="text/javascript" src="js/jquery.js"></script>
13 <script type="text/javascript" src="js/opacity.js"></script>
14 </body>
15 </html>
```

Figure-54.Source code 32

We simply create two images and assign them class="fadeto".

Next we need to edit our "js" file as below:

```
1  $(document).ready(function(){
2   $('.fadeto').css('opacity','0.4');
3  $('.fadeto').mouseover(function(){
4  $(this).fadeTo(100,1,function(){
5   $('.fadeto').not(this).fadeTo(100,0.4);
6   });
7   });
8   });
```

Figure-55.Source code 33

The code in "js" file means that in normal state opacity of pictures is 0.4. When we hover over one of them, its opacity becomes 1. And when we hover over another one, its opacity goes back to 0.4.

Here is how it looks:

Figure- 56.Example image 12

(Alex Garrett, Gallery fading effect, 2011)

Adding to dropdown

Garret defines that "adding to dropdown" is a small plugin that is very interesting and useful. It is usually seen in a websites with questionnaires and polls. Fundamentally, it adds to a drop down menu the item that we select.

In our example we will take a look at that function.

First of all we need to create a "js" file and call it "dropdown.js". Then let's connect it to our "index.php" file. Then let's edit our "index.php" file as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<p>
<a href="#" class="link">Finland</a>
<a href="#" class="link">Sweden</a>
<a href="#" class="link">Norway</a>
<a href="#" class="link">Iceland</a>
</p>
<select id="list"></select>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/dropdown.js"></script>
</body>
</html>
```

Figure-56.Source code 34

We mainly added four words with links that can be chosen. Then we created an empty dropdown.

Now it is time to modify our “js” file.

```
$('.link').click(function(){
var item=$(this).text();
$('#list').append('<option>'+item+'</option>');
});
```

Figure-57.Source code 35

This code mainly means that we add a chosen word to our drop down.

Here is how it looks like:

Figure- 58.Example image 13

(Alex Garrett, Adding to a dropdown, 2011)

Scroll to top

Nowadays, we can see many new popular plugins all over the internet. They are very useful and interesting. jQuery is meant to make our "journey" in the web easier and smoother.

One of these kinds of plugins is "scroll to top". When we read huge articles, and we get to the end, we see a link saying "scroll to top" or "back to top". When we press it we go to the beginning of the page. Next example will do the same trick.

To customize this kind of plugin we need first to create a "js" file and call it "scrollto-top.js". Then we connect it to our "index.php" file. Then it is time to edit our "index.php" file and add there a hug text and a button with "href".

```
In 1918 during the Civil War, the headquarters of the White army were established at the Hotel Seurahuone in Mikkeli. Mikkeli was located in a
During the Winter War and Continuation War, the headquarters of the Finnish Army was located in Mikkeli.[13] The Army staff made their base in
Wartime Mikkeli is identified with Marshal Mannerheim, the commander of the Finnish army and later President of Finland. His personal railway ca
One of the main museums in the town is the Infantry Museum (Jalkaväkimuseo) located in one of the former army barracks, close to the University
In 1997 there was a province reform, which made Mikkeli the capital of the new province of Eastern Finland. In a separate reform, the rural mun
[edit]Trivia.The former and present Finnish commissioners, Erkki Liikanen and Olli Rehn, are both born Mikkelians.
The local football team is the First Division Mikkelin Palloilijat (MP), for which Olli Rehn used to play for 13 years (youth teams 1968–78, fi
On August 9, 1986, there was a siege in Mikkeli after a bank robbery that had taken place the previous day in Jakomäki, Helsinki. The siege lea
[edit]Sport.Jukurit is an ice hockey team in Mikkeli. The team has won four Mestis's championships (2001, 2002, 2003 and 2006).
Mikkelin Kampparit, or just Kampparit, plays in the highest bandy division.[14] In 2012 they became Finnish champions for the first time.[15
Mikkeli (ruots. S:t Michel) on Suomen kaupunki ja Etelä-Savon maakuntakeskus, joka sijaitsee Saimaan luoteisrannalla Etelä-Savon maakunnassa.
Kaupungissa asuu 48 964 henkilöä[2], ja sen pinta-ala on 2 124,60 km², josta 424,92 km² on vesistöjä. Järviä ja lampia kaupungissa on noin 700.
Mikkelin naapurikunnat ovat Hirvensalmi, Juva, Kangasniemi, Pieksämäki, Puumala ja Ristiina.
Mikkeli sijaitsee Vuoksen vesistön ja Kymijoen vesistön välisellä vedenjakajalla. Kaupungin keskusta on Saimaan Savilahden rannalla. Kaupungin
Mikkelin kaupunginjohtaja on Kimmo Mikander (kok.).
Vanhimmat löydetyt merkit asutuksesta Mikkelin seudulla ovat kivikauden kampakeraamiselta kaudelta  4000–2000 eaa. Tältä aikakaudelta tunnetut
Vanhimmat kirjalliset maininnat nykyisen Mikkelin seudun asutuksesta ovat vuodelta 1323 Pähkinäsaaren rauhansopimuksesta, jolla Savilahden pogos
Saksalaissiirtokunnan jäseniä muutti Tarsalanjärven rannalle vuonna 1540, silloiseen Vesulahden (Visulahden) pitäjään.
Mikael Agricola kävi tarkastuskäynnillä Savilahdella 1549.
Nuijasodassa (1596–1597) surmattiin yli 200 talonpoikaiskapinallista nuijamiestä Mikkelin pitäjän pappilassa Kenkäverossa 1597.
Kustaa III:n sodassa (1788–1790) käytiin vuonna 1789 muutamia kilometrejä Mikkelin kirkonkylästä etelään Porrassalmen taistelu, jossa ruotsalais
Mikkeli sai kaupunginoikeudet 1838 Venäjän keisari Nikolai I:ltä. Vuonna 1831 perustetun Mikkelin läänin lääninhallitus siirtyi Heinolasta Mikk
Sisällissodassa (1918) Mikkeli oli yksi paikkakunnista, joilta hallituksen, Suomen tasavallan joukkojen eli valkoisten toimintaa johdettiin. To
Mikkeli on ainoa suomalainen paikkakunta, jolle on myönnetty Vapaudenristi – 4. luokan Vapaudenristi miekkoineen – vuonna 1944 kiitoksena pääma
Elokuussa 1986 Mikkelin torin laidalla räjähti poliisin piiritystilanteessa auto, jolla edellisenä päivänä Kansallis-Osake-Pankin konttorin Hel
Lääniuudistuksessa 1997 Mikkelistä tuli Itä-Suomen läänin pääkaupunki ja läänien lakkauduttua Itä-Suomen aluehallintoviraston sekä Etelä-Savon
Mikkeli, Anttola ja Mikkelin maalaiskunta ja yhdistyivät vuonna 2001 uudeksi Mikkelin kaupungiksi, johon liitettiin Haukivuori
<p><a class="top" href="#">Go to top</a></p>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/scroltotop.js"></script>
</body>
</html>
```

Figure-59.Source code 36

Now it is time to write our code to "js" file.

```
1  $(document).ready(function(){
2  $('.top').click(function(){
3  $('html,body').animate({scrollTop:0},1000);
4  });
5  });
```

Figure-60.Source code 37

This code mainly tells to go back to top with speed “1000” whenever we press a button.

Here is how it works:

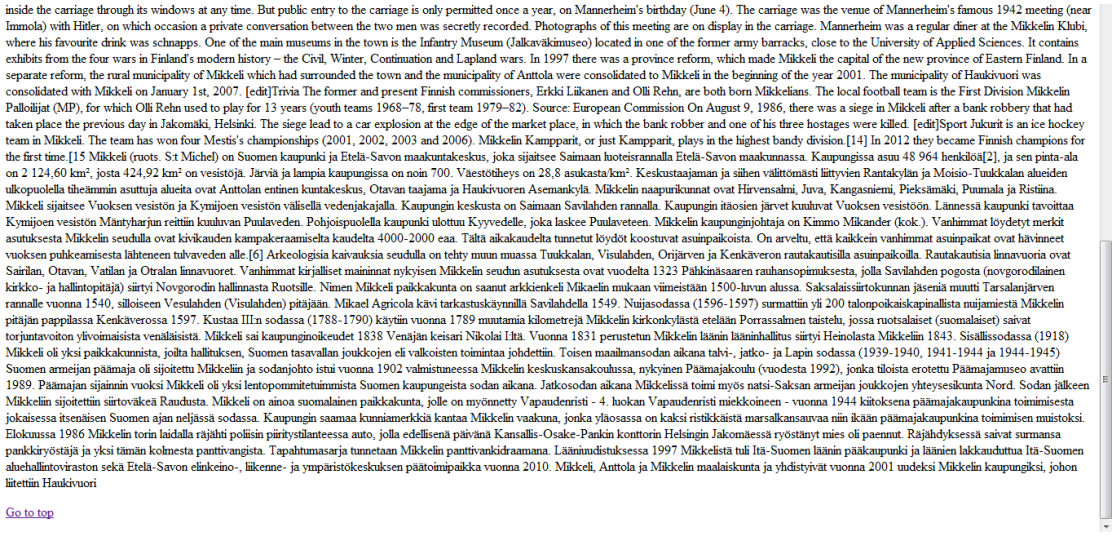


Figure- 61.Example image 14

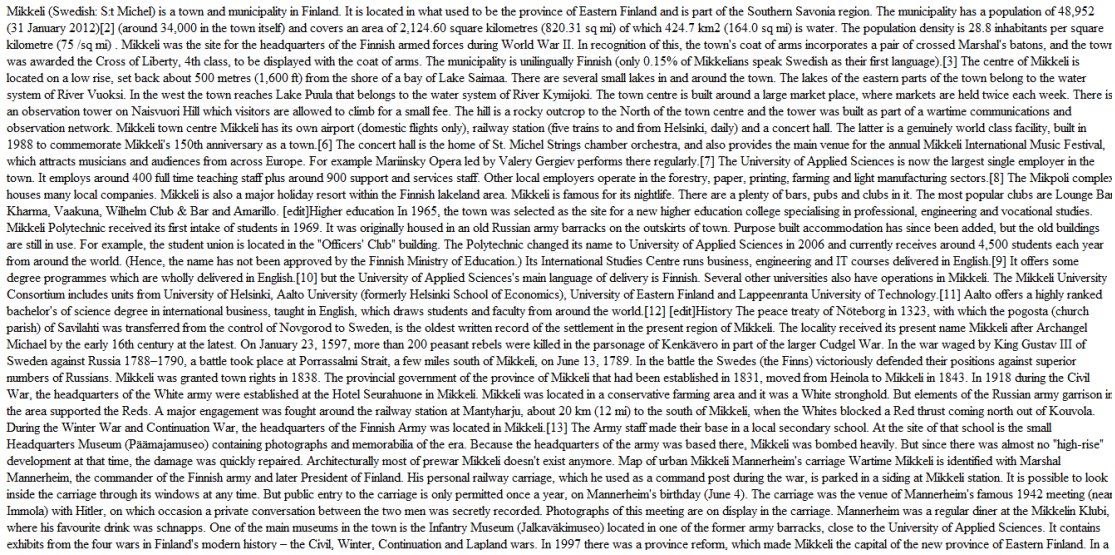


Figure- 62.Example image 15

On the **Figure- 61** we see the button “Scroll to top”. And when we press it we go up to top, as it shows on the **Figure- 62.** (Alex Garrett, Scroll to top, 2011)

Enable a checkbox after scroll

All of us have agreed for “Terms and Conditions”. Not all of us read them usually. However, it is very important to read them, because they contain valuable information.

Lately, in “Terms and Conditions” “scroll to enable a checkbox” plugin is used. This plugin mainly does not allow you to agree for “Terms and Conditions” if user did not scroll till the end.

In next example we will see this on practice.

First of all, as usual, we need to create a “js” file and call it “enableckeck.js”. Then let’s connect it to “index.php”.

Now let’s edit our “index.php” as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<p><textarea id="terms" rows="10">This Statement of Rights and Responsibilities ("Statement," "Terms,"
<p><input id="agree" type="checkbox"/>I agree to the terms and conditions</p>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/enablecheck.js"></script>
</body>
</html>
```

Figure-63.Source code 38

Here we mainly create a “textarea” and add text to it. Then we create a textbox.

Now we will edit “js” file and write our code.

```
$(document).ready(function(){
$('#agree').attr('disabled','disabled');

$('#terms').scroll(function(){
var textarea_height=$(this)[0].scrollHeight;
var scroll_height=textarea_height- $(this).innerHeight();
var scroll_top=$(this).scrollTop();
if (scroll_top==scroll_height){
$('#agree').removeAttr('disabled');
}
});
});
```

Figure-64.Source code 39

This code means that a textbox with id=”agree” is disabled until the mouse scrolls till then end. Then it activates.

Here how it works:

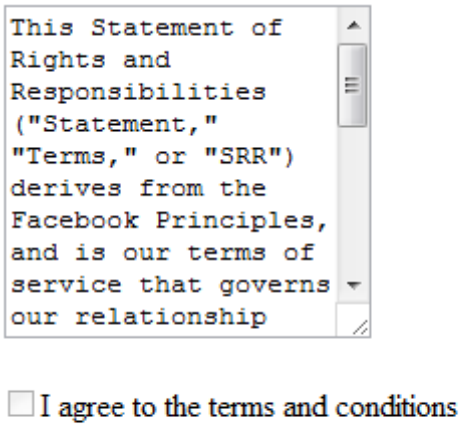


Figure- 65.Example image 16

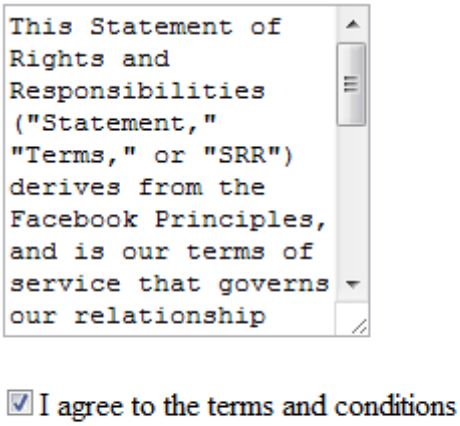


Figure- 66.Example image 17

(Alex Garrett, Enabling checkbox after scroll, 2011)

Show password

Garret says that plugins are mainly ready files that work after connecting them to "index.php" file. jQuery website has a link where it is possible to download ready plugins, customized by users.

One of these kinds of plugins is "show password".

Sometimes on some websites where we either log in or sign up, we type passwords. However, we usually face a problem when we do not know what we type. To avoid mistakes we press a button called "show password".

In the next example we will take a look at that plugin.

First of all we need to create a “js” file and call it “password.js”. Then we need to download a pack of “show password” plugin which consist of two “js” files. This pack can be downloaded from link below:

https://www.dropbox.com/s/jftjl6m0atshc59/jquery.showpassword-1.3.zip

Next we connect both, “js” file from pack and “password.js” to our “index.php” file.

Now it is time to modify our “index.php” file.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<div class="form">
            <p>
                <label>Password</label>
                <span class="w">
                    <input id="password" type="password" data-typetoggle='#show-password' class="input" />
                    <label><input id="show-password" type="checkbox" /> Show password</label>
                </span>
            </p>

        </div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/showpassword.js"></script>
<script type="text/javascript" src="js/password.js"></script>
</body>
</html>
```

Figure-67.Source code 40

Here we mainly add one input box and one check box.

Next we modify “password.js” file as below:

```
$(document).ready(function(){
                $('#password').showPassword();
            });
```

Figure-68.Source code 41

In these lines of code we mainly assign an item with id “password” which is an input box to a function called “showPassword”.

Here is how it looks:

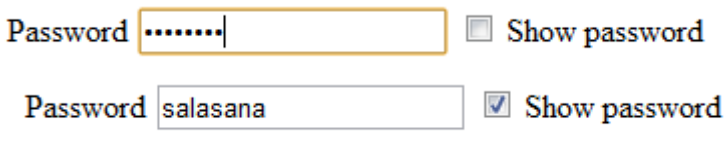


Figure- 69.Example image 18

(Alex Garrett, Show password, 2011)

Learning jQuery UI (User Interface)

jQuery has a wide range of functions that are very useful and interesting. jQuery UI is one of the tools of jQuery which is really easy to work with and is very attractive.

jQuery UI allows us to implement plugins that we cannot make with jQuery. It is really easy to implement and use.

jQuery UI has interactions as :

- Draggable
- Droppable
- Resizable
- Selectable
- Sortable

And also widgets as:

- Accordion
- Autocomplete
- Button
- Datepicker
- Dialog
- Progressbar
- Slider
- Tabs

(Alex Garrett, Learning jQuery UI (User Interface), 2011)

Installing jQuery UI

Installation of jQuery UI is very easy. All we need to do is to go to the official website which is http://jqueryui.com. There we download a pack that contains many useful items.

First step in installation is to extract files from zip archive. From folder "js" we take a file called "jquery-ui-1.8.24.custom.min.js" and copy it to our "js" folder. Then we rename it to "jquery-ui.js". Next we copy a folder called "development-bundle" to our root directory, where our "index.php" file is. Then we copy a folder called "ui-lightness" to "css" folder in our root directory. That is it, jQuery UI is installed.

Next step is testing jQuery UI. (Alex Garrett, Installing jQuery UI, 2011)

Testing jQuery UI

To test jQuery UI we need to customize a simple plugin and see if it works.

Let's use a draggable interaction in this plugin.

First of all we include "jquery-ui.js" in our "index.php" file. Then we create a "js" file and call it "uitest.js" and also connect it to "index.php".

Now we need to modify "index.php" file as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<span id="drag">You can actully drag this!</span>

<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/uitest.js"></script>

</body>
</html>
```

Figure-70.Source code 42

Here we mainly created a “span”, gave it an id=”drag” and wrote some text.

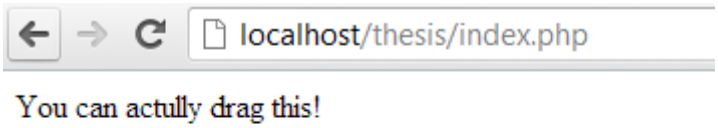
Next we modify “uitest.js” as below:

```
$(document).ready(function(){
$('#drag').draggable();
});
```

Figure-71.Source code 43

This code means that when the page is fully loaded, we can drag a text that we assign to a span.

Here is how it works:



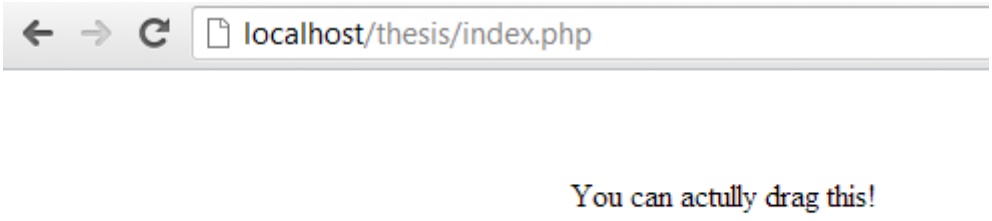


Figure- 72.Example image 19

(Alex Garrett, Testing jQuery UI, 2011)

Droppable

Droppable interaction is very interesting and alluring function of jQuery UI. We do not see it often nowadays in simple websites, but we can see that much in some online games etc. The basic idea of "droppable" is that we can drag an item and drop it somewhere.

In next example we will take a look at a situation where we drag a text and drop it in a box, and it changes.

First step is to create a "js" file and call it "drop.js". Then we include it to our "index.php" file.

In this example, we also use data from the previous example. Essentially, we modify previous example and make some additions to it.

After including "js" file to our "index.php", we modify it as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf=8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<span id="drag">You can actully drag this!</span>
<div id="drop"></div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/drop.js"></script>

</body>
</html>
```

Figure-73.Source code 44

Essentially, all we did here is, we just added "div" with an id="drop".

Then we modify "drop.js" file as below:

```
$(document).ready(function(){
$('#drag').draggable({containment:'document',revert:false});
$('#drop').droppable({hoverClass:'border'});
});

```

Figure-74.Source code 45

These lines of code mean that item with id=”drop” is droppable and with id=”drag” is draggable. And droppable item has a property of hoverClass, which is called “border”. Next step is to modify “style.css” file and change outlook of “div” and “border”.

```
#drop{
height:200px;
width:200px;
background-color:#f0f0f0;

}
.border{
border:1px solid #000;
}
```

Figure-75. Source code 46

Here we mainly gave an outlook to “drop” and “border”.

Here is how our new plugin looks like:

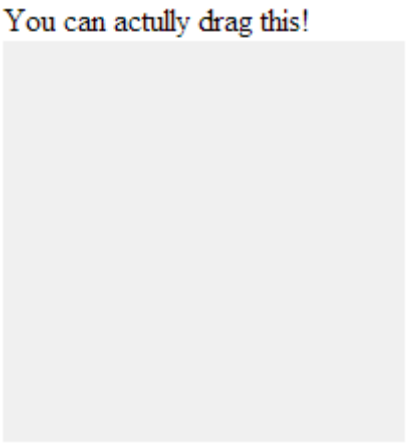


Figure- 76. Example image 20

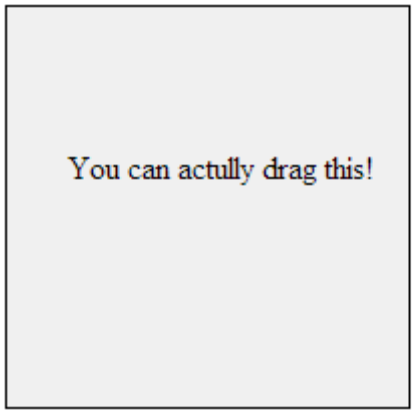


Figure- 77.Example image 21

(Alex Garrett, Droppable, 2011)

Sortable

Next useful tool in jQuery UI is “sortable” interaction. This allows us to sort a list of items in any order we want. It has many properties.

In our example we will take a look at simple “sortable” interaction of a list of names.

To start, we need to create a “js” file and call it “sort.js” and then include it to our “index.php” file. Next we need to edit our “index.php” file as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf=8">
<title>jQuery Tutorial</title>

<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<ul id="names">
<li>Matti</li>
<li>Jukka</li>
<li>Ville</li>
<li>Petri</li>
</ul>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/sort.js"></script>

</body>
</html>
```

Figure-78.Source code 47

Here we mainly created a list of names and gave it an id=”names”.

Now it is time to modify the “style.css” file.

```
ul{
width:200px;
padding:0px;
list-style:none;


}
```

Figure-79.Source code 48

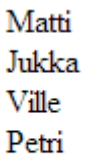
Next step is to edit and write our code to “sort.js” file:

```
1  $('#names').sortable({containment:'parent',tolerance:'pointer',cursor:'pointer',revert:'true'});
2
```

Figure-80.Source code 49

This code mainly means that the item with an id=”names” is sortable. Containment means simple properties of that interaction.

Here is how our final version looks like:



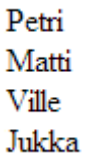


Figure- 81.Example image 22

On the last image we see how we can easily sort a list by just dragging it over another item. (Alex Garrett, Sortable, 2011)

Resizable

“Resizable” interaction is very important part of jQuery UI. It mainly does what it says, allowing us to resize an element which in other cases cannot be resized. In our example we will create an element and then resize it as we want.

First of all we need to create a “js” file and call it “resize.js”. Now let’s include it into our “index.php” file. Also, we need to include a “css” file from “css/ui-lightness” called “jquery-ui-1.8.24.custom.css”. In other cases this file can be called differently, depending on the version of package.

Now we modify “index.php” file.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<div id="box"></div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/resize.js"></script>

</body>
</html>
```

Figure-82.Source code 50

Here we mainly created a “div” and gave it an id=”box”. Now we need to style this “div” in ”style.css”.

```
#box{
width:100px;
height:100px;
background-color:#999;
border:1px solid #000;

}
```

Figure-83.Source code 51

We changed its width, height and gave it a background color and border.

Now it looks like this:

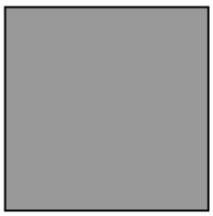


Figure- 84.Example image 23

Now it is time to write our code to “js” file.

```
$('#box').resizable();
```

Figure-85.Source code 52

This code essentially allows the box to be resizable.

Here is what we get as a final result:

Figure- 86.Example image 24

(Alex Garrett, Resizable, 2011)

Accordion

"Accordion" is very interesting plugin. Accordion originally is a musical instrument. The basic idea of jQuery plugin "accordion" is that it looks like that musical instrument, it have many levels which open when we press it.

In our next example we will see how it works.

First, we need to create a "js" file and call it "accordion.js" then we include it to our "index.php" file. Next we modify "index.php" file as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<div id="content">
<h3><a href="#">Suomi</h3>
<div>Finland</div>


<h3><a href="#">Ruotsi</h3>
<div>Sweden</div>



<h3><a href="#">Norja</h3>
<div>Norway</div>

</div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/accordion.js"></script>

</body>
</html>
```

Figure-87.Source code 53

We mainly created "divs" and "headers" inside one large "div" with an id="container".

Next we need to modify "js" file.

```
1    $('#content').accordion();
```

Figure-88.Source code 54

With only 1 line of code we made a magic. Now from these “divs” we get an accordion.

Here is how it looks:

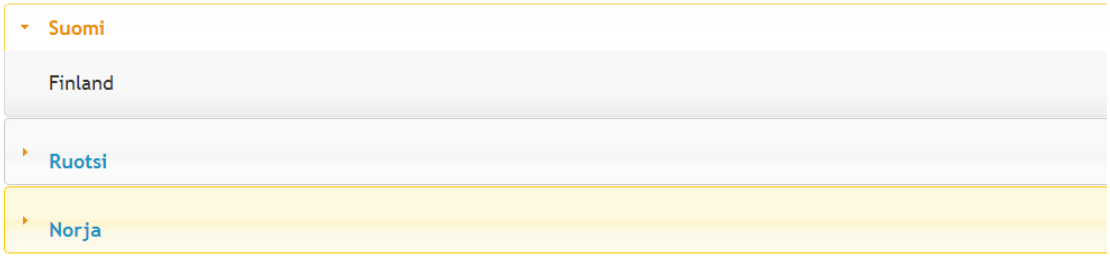


Figure- 89.Example image 25

(Alex Garrett, Accordion, 2011)

Datepicker

Garret claims that “Datepicker” is a very useful tool that allows us to pick a date on websites. We can use this plugin to pick dates of birth, graduation, or some deadline. This tool is widely used all over the internet. Although it looks very professional, it is extremely easy to implement.

To do that, we need to first create a “js” file and call it “date.js” and include it in our “index.php” file. Then we add some items to “index.php” file.

```
1   <!doctype html>
2   <html lang="en">
3   <head>
4   <meta charset="utf-8">
5   <title>jQuery Tutorial</title>
6   <link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
7   <link rel="stylesheet" type="text/css" href="css/style.css" />
8   </head>
9   <body>
10  <input id="date" type="text" size="8"/>
11  <script type="text/javascript" src="js/jquery.js"></script>
12  <script type="text/javascript" src="js/jquery-ui.js"></script>
13  <script type="text/javascript" src="js/date.js"></script>
14
15  </body>
16  </html>
```

Figure-90.Source code 55

Here we mainly added an input box and gave it an id=”date”.

Next we need to change a font size in “style.css” file.

```
1  body{
2    font-size:10px;
3
4  }
```

Figure-91.Source code 56

Now it is right time to write our code in “date.js”.

```
1  $('#date').datepicker();
```

Figure-92.Source code 57

This code mainly means that when we press the “date” input box, “datepicker” appears.

Here is how it looks like:

Figure- 93.Example image 26

(Alex Garrett, Datepicker, 2011)

Progress bar

“Progress bar” is a very nice and important tool in jQuery. “Progress bar” is a bar that shows how many percent are uploaded or downloaded.

In our example we will not customize a plugin in which we will upload or download a file, we will just simulate the situation when we press a button and the process bar goes up to a certain value.

First of all we need to create a “js” file as usual, and call it ”progress.js”. Then we need to include it to our “index.php” file.

Next we need to edit “index.php” file as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<input id="upload" type="button" value="Upload"/>
<div id="percent"></div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/progress.js"></script>

</body>
</html>
```

Figure-94.Source code 58

Here we mainly created a button with an id=”upload” and a “div” with an id=”percent”.

Next step is to go to “style.css” and modify it a bit.

```
body{
font-size:12px;

}
#percent{
width:400px;
}
```

Figure-95.Source code 59

Next is our “js” code in “progress.js” file:

```
$('#upload').click(function(){
var val=0;
var interval=setInterval(function(){
val=val+1;
$('#percent').progressbar({value:val});
},50);

});
```

Figure-96.Source code 60

Here we mainly simulated a situation when something is being uploaded. So when we press upload button the percentage starts to grow.

Here is how it works:

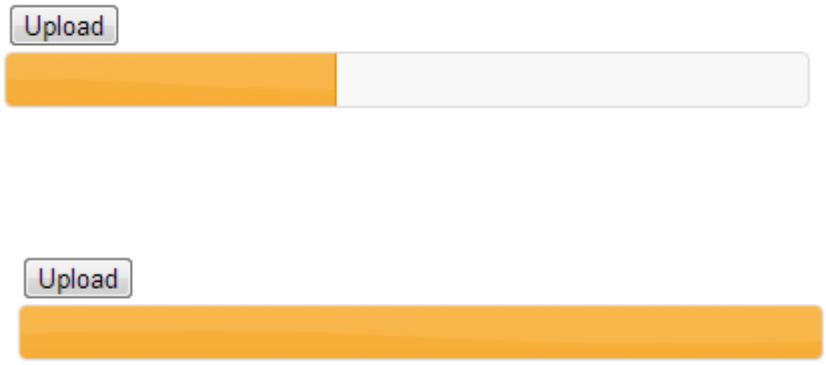


Figure- 97.Example image 27

(Alex Garrett, Progress bar, 2011)

Slider

One of the nicest and simplest plugins is “slider”. It is used widely all over the internet. Commonly we can see this plugin in registration websites where we need to choose a certain value.

Fundamentally, the slider works so that, it has a bar which can be slided, and while that the value changes.

Our next example will demonstrate that.

First of all we need to create a “js” file and call it “slider.js”. Then we modify “index.php” and include it there. Next step is to add some lines of code to “index.php”.

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<div id="slider"></div>
<div id="slider_value"></div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/slider.js"></script>

</body>
</html>
```

Figure-98.Source code 61

Here we mainly created two “div”s with ids.

Next we modify “style.css” file as below:

```
body{
font-size:14px;
}
#slider{
width:400px;

}
```

Figure-99.Source code 62

And the last step is to write our code to "slider.js" file:

```
$('#slider').slider({
slide: function(event, ui){
$('#slider_value').html('&pound;'+ui.value);
}

});
```

Figure-100.Source code 63

These lines of code mainly mean that when we slide, the value goes up and the pound sign is added to it.

Here is how it all looks like:

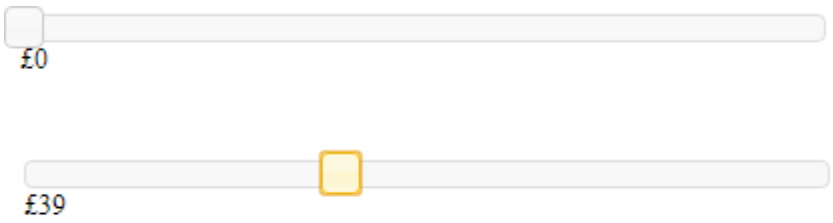


Figure- 101.Example image 28

(Alex Garrett, Slider, 2011)

Tabs

Tabs are very similar to horizontal menu. They look very attractive and are very easy to implement. Essentially, they show some item when they are pressed.

In our example, we will consider a situation when a tap shows us a text when we press it.

First of all in this example we need to create a "js" file and call it "tabs.js".

Now we need to include it to our "index.php" file and modify it as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<div id="tabs">
<ul>
<li><a href="#suomi">Suomi</a></li>
<li><a href="#ruotsi">Ruotsi</a></li>
<li><a href="#norja">Norja</a></li>
</ul>
<div id="suomi">
<p>Finland</p>
</div>
<div id="ruotsi">
<p>Sweden</p>
</div>
<div id="norja">
<p>Norway</p>
</div>
</div>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/tabs.js"></script>

</body>
</html>
```

Figure-102.Source code 64

Next we modify “style.css” file as below:

```
body{
font-size:14px;
}
#tabs{
width:400px;

}
```

Figure-103.Source code 65

We mainly created a list with “href”. The “href” leads us to a “div” with certain text. Now we need to write our code to “js” file.

```
$('#tabs').tabs();
```

Figure-104.Source code 66

The code is very short and simple. It mainly transforms item with id=”tabs” into real “tabs” tool.

Here is how it looks:

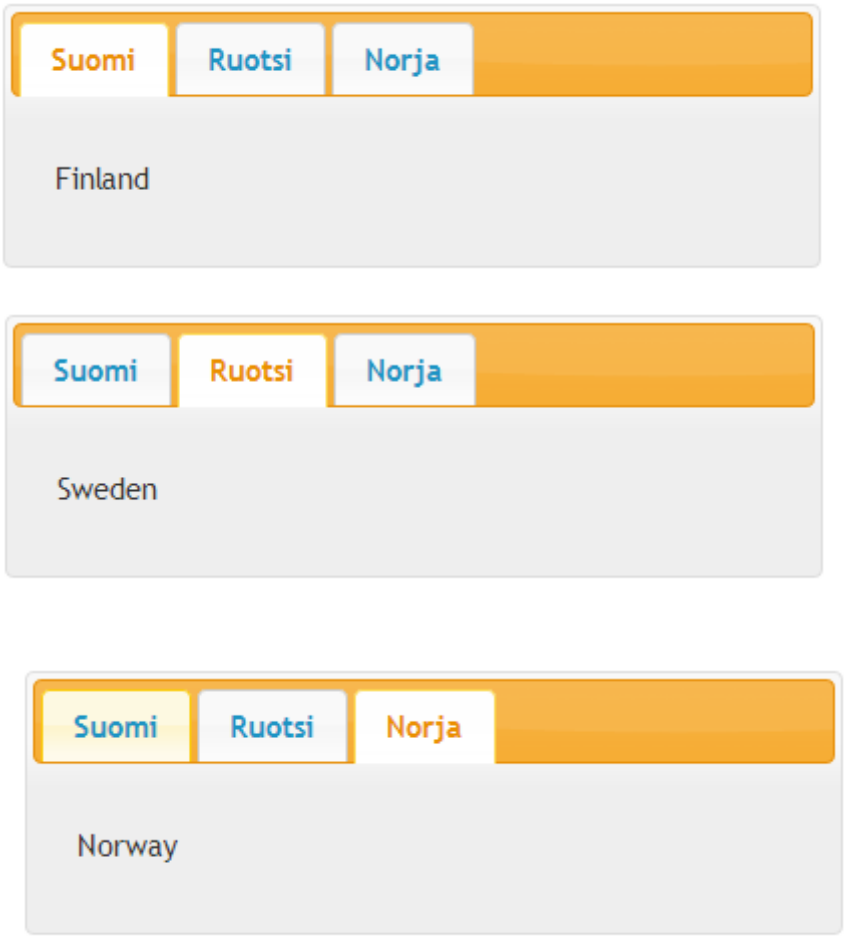


Figure- 105.Example image 29

(Alex Garrett, Tabs, 2011)

Drag and drop list

Previously, we reviewed “droppable” and “draggable” interactions, but the next plugin will contain them both. “Drag and drop” list is a very interesting and simple plugin that can work as a small application. We can use it to make a game, to-do list application and so on.

In the example we will review a plugin which adds items from original list to our “to-buy” list.

First of all, as usually, we need to create a “js” file and call it “dragdrop.js”. Then we include it in “index.php” function and modify it as below:

```html
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<ul>
<li class="item">Suomi</li>
<li class="item">Ruotsi</li>
<li class="item">Norja</li>
<li class="item">Tanska</li>

</ul>
<div id="list"></div>

<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/dragdrop.js"></script>

</body>
</html>
```

Figure-106.Source code 67

We created list and “div” and gave them id.

Next we need to style this “div” and list in “style.css” file.

```css
ul{
width:100px;
padding:0px;
list-style:none;
}
#list{
width:120px;
height:180px;
background-color:#f0f0f0;
border:1px solid #000;

}
#list.border{
border-width:2px;
}
```

Figure-107.Source code 68

Now we need to write our code in “js” file.

```
1  $(document).ready(function(){
2  $('li').draggable({containment:'document',revert:true,
3  start:function(){
4  contents=$(this).text();
5
6  }
7  });
8  $('#list').droppable({hoverClass:'border',accept:'.item',
9  drop:function(){
10 $('#list').append(contents +'<br/>');
11 }
12 });
13 });
```

Figure-108.Source code 69

These lines of code mean that when we drag and drop an item from our list the “div” box, the stay there, and an item goes back to its place.

Here is how it works:

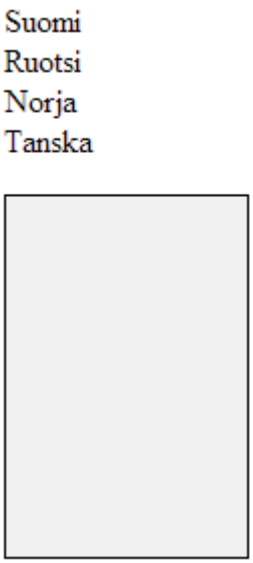


Figure- 109.Example image 30

Figure- 110.Example image 31

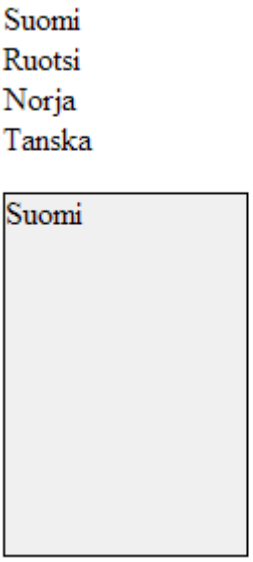


Figure- 111.Example image 32

(Alex Garrett, Drag and drop list, 2011)

Slide down message

"Slide down message" is a very important and interesting plugin. It is very common nowadays on internet. We can see this plugin almost on every site.

It is also easy to use and implement.

To implement it we need to create two "js" files and call them "button.js" and "slidemessage.js".

Then we include them to our "index.php" file and edit it as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<div id="slideNotice">  </div>
<h3>Settings</h3>
<p>User information</p>
<p><input id="save" type="button" value="Save" /></p>

<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/button.js"></script>
<script type="text/javascript" src="js/slidemessage.js"></script>

</body>
</html>
```

Figure-112.Source code 70

Here we mainly created one “div” with an id=”slideNotice” and paragraphs and button.

Next is “style.css” file.

```
#slideNotice{
display:none;
position:absolute;
height:50px;
top:0;
left:0;
width:100%;
background-color:#f0f0
text-align:center;
}
#slideNotice h3{
posistion:relative;
padding:0px;
margin:0px;
top:25%;

}
```

Figure-113.Source code 71

Now we need to modify “js” files and write our code in there.

```
function slideNotice(text){
$('#slideNotice').html('<h3>'+text+'</h3>').slideDown();

}
```

Figure-114.Source code 72

```
1  $('#save').click(function(){
2  slideNotice('Your setting are saved!');
3
4  });
```

Figure-115.Source code 72

This code means that when we press “save” button we call function “slideNotice”. Also we implemented appearance method which is “slideDown”.

Here how it looks like this:

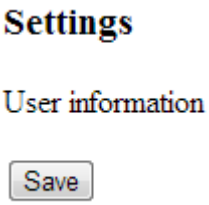


Figure- 116.Example image 33

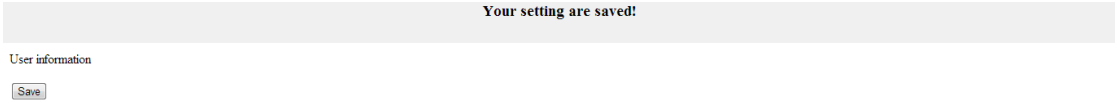


Figure- 117.Example image 34

(Alex Garrett, Slide down message, 2011)

Smileys icons

Lately, smileys have become very important in our lives. Every device already has a built in pack of fancy and funny emoticons that allow people to express their feeling using text.

Emoticons nowadays are very different. Some are just simple pictures, others are animations. Google talk has a very interesting type of smileys for example: they are animated, but they consist of keyboard characters.

In our example we will review a plugin that allows us to add smileys to our text. Usually this plugin works with pictures, but in our example we will just use keyboard characters. The main idea is that we retrieve value of the button and add it to a text area.

To start with this plugin, we need to create a “js” file and call it “smiley.js”. Then we include it to our “index.php” file and edit it as below:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>jQuery Tutorial</title>
<link rel="stylesheet" type="text/css" href="css/ui-lightness/jquery-ui-1.8.24.custom.css" />
<link rel="stylesheet" type="text/css" href="css/style.css" />
</head>
<body>
<p> Leave a comment here:<br/><textarea id="comment"></textarea></p>
<p>
Add smiley :
<input class="smiley" type="button" value=":)"/>
<input class="smiley" type="button" value=":D"/>
<input class="smiley" type="button" value=":("/>
<input class="smiley" type="button" value=";)"/>
<input class="smiley" type="button" value=":O"/>
<input class="smiley" type="button" value=":*"/>
</p>
<script type="text/javascript" src="js/jquery.js"></script>
<script type="text/javascript" src="js/jquery-ui.js"></script>
<script type="text/javascript" src="js/smiley.js"></script>
</body>
</html>
```

Figure-118.Source code 73

Here we mainly created “textarea” with id=”comment” and buttons with values.

Next we modify “style.css” to make a “textarea” look better.

```
#comment{
height:90px;
width:360px;
}
```

Figure-119.Source code 74

Next and last step is to write our “js” code in “smiley.js” file.

```
$(document).ready(function(){
$('.smiley').click(function(){
var textarea_val=$('#comment').val();
var smiley_val=$(this).attr('value');
$('#comment').val(textarea_val+' '+smiley_val);
});
});
```

Figure-120.Source code 75

With these lines of code we mean that when a curtain button is clicked the value of it is added to our “comment textarea”.

Here is how it looks:

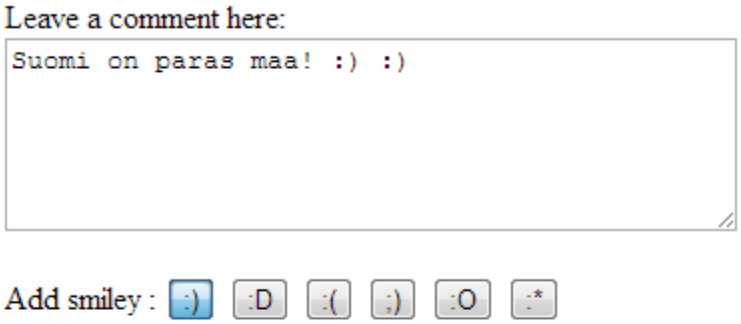


Figure- 121.Example image 35

(Alex Garrett, Emoticons, 2011)

This is the end of the theoretical part, where jQuery learning methods were discussed. These methods are standing on a basis of the practical part which will be discussed in the next chapter.

4 PRACTICAL IMPLEMENTATION

4.1 Methodology

In this chapter I will review modules that were developed using all the knowledge I gained in previous chapters.

Theoretical section consists of two parts:

- Customizing jQuery plugins
- Converting plugins into Joomla modules

All together I have developed plugins listed below:

- Language switcher
- Slide-in box
- Slide-up menu
- Drop-down menu

NOTE:

Later on in codes we will face lines like:

```
<?php echo $Something;?>">
```

This line of code is a reference to a user interface of Joomla. We need it to modify the items through a user interface in Joomla CMS.

Example:

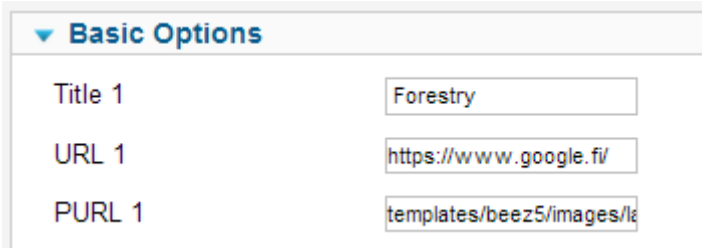
```
<a href="https://www.google.fi/">
```

This code is fixed. It means that later on if we create a plugin it will have the same value and if an inexperienced user will need to modify a plugin he will need to access the code.

```
<a href="<?php echo $url1;?>">
```

This code will allow us modify the code from Joomla GUI window.

It will look like this:



4.2 Language switcher

Nowadays we live in a global world where websites are translated into many languages. Not often we can see a website that supports only one language.

The most common way of switching language is pressing on a flag of the country. For example, if we want to choose British English we press on British flag.

However, we can make it look much better using jQuery.

This is probably the easiest plugin to make among others.

The idea is to make a drop down list with a link and modify it to look as we want.

First of all we need to change our strategy. The idea is that, later on we will need to convert our plugins into Joomla supportable modules. Therefore we can ease our task starting from now.

First of all let's modify our "index.php" file and call it "default.php".

Now we can add some lines of code to it.

First step is to create a "div" where the switcher will appear. Let's call it "country-select". Then we create a "form". We leave it empty.

Next step is to create a "select" and give it an id="country-options".

Now we need to include those options. We will have as many options as we want which equals to number of languages we want to have on our webpage.

Last step is to include "js" file which we should create now and call "languageswitcher.js"

Also, we should include all external files to "default.php", such as "css" and "js" files.

All together our “default.php” file should look like this :

```
<html >
<head>
<link rel="stylesheet" type="text/css" href="modules/mod_language_switcher/css/languageswitcher.css" />
<div id="country-select">
<form >
<select id="country-options" name="country-options">
        <option  <?php echo $english;?> title="/mhgsystems.com/index.php/en/" value="us">English</option>
        <option  <?php echo $finnish;?>  title="/mhgsystems.com/index.php/fi/" value="fi">Suomea</option>
        <option <?php echo $russian;?>  title="/mhgsystems.com/index.php/ru/" value="ru">Русский</option>
        <option <?php echo $chinese;?> title="/mhgsystems.com/index.php/zh/" value="zh">中国的</option>
        <option <?php echo $spanish;?> title="/mhgsystems.com/index.php/es/" value="nl">Español</option>
</select>
<script src="modules/mod_language_switcher/js/languageswitcher.js"></script>
</head>
</html>
```

Figure-122.Source code 76

Now we need to modify “style.css” file as below:

(As the code is too long, print screen will be hard to see. Consequently the code will be pasted into appendices.)

Next step is to create a “js” file and call it “languageswitcher.js”. The code will be placed into appendices.

Here is how our final module will look like:

Figure- 123.Example image 36

4.3 Slide-in box

Nowadays we can usually see websites with long articles, usually they are news websites. While we read those articles some ads pop up. This is usually annoying and irritating.

However, lately new plugins have come out which slide in a box from the bottom without interrupting any reading action.

Our example does the same function. It is based on jQuery. It is called “slide-in box”.

To start with this plugin we need again create a “default.php” file and modify it.

First of all we create a “div” and assign it an id=”slidein”. Then we create an item and give it a class=”close” for a close button. Then we write down a content of a box in

"div" that we created. Also we include "css" and "js" files to it. It all should look like this:

```
<link rel="stylesheet" type="text/css" href="modules/mod_slidein/css/style.css" />
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1/jquery.min.js"></script>
<script type="text/javascript" src="modules/mod_slidein/js/slidein.js"></script>
<div id="slidein" >
<a class="close"></a>
<h1> <?php echo $Heading;?></h1>
<h2><?php echo $Text;?></h2>
<br><br>
<a  class="more" href=<?php echo $URL;?> ></a>
</div>
```

Figure-124.Source code 77

Next step is "css" file.

1st part	2nd part

#slidein{
background-image:
url(/mhgsystems.com/templates/beez5/images/layout/cubes.png);
text-align:center;
width:350px;
height:100px;
padding:10px;
position:fixed;
bottom:0px;
border-top:3px solid #0000;
right:-430px;
-moz-box-shadow:1px 1px 6px #000;
-webkit-box-shadow:1px 1px 6px #000;
-moz-border-radius:0px 0px 10px 10px;
-webkit-border-top-left-radius:5px;
-webkit-border-bottom-left-radius:5px;
border-radius:10px 10px 0px 10px;
}
#slidein p, a.more{
font-size:14px;
text-transform:uppercase;
font-family: Arial,Helvetica,sans-serif;
letter-spacing:1px;
color:#555;
}
#slidein h1{
line-height:140%;
align:center;
color:#FF9933;
font-size:14px;
margin:10px 20px 10px 0px;
}
	#slidein h2{
	line-height:140%;
	align:center;
	color:#333333 ;
	font-size:14px;
	margin:10px 20px 10px 0px;
	}
	a.close{
	background:transparent
	url(http://localhost/latest/templates/beez5/images/layout/close.png)
	no repeat top left;
	width:22px;
	height:22px;
	position:absolute;
	cursor:pointer;
	top:13px;
	right:13px;
	}
	a.more{
	background:transparent
	url(http://localhost/latest/templates/beez5/images/layout/button_read
	more.png) no-repeat top left;
	width:120px;
	height:50px;
	position:absolute;
	cursor:pointer;
	top:83px;
	right:130px;
	}
	a.close:hover{
	#header{
	font-size:100px;
	}
	}

Table-1

The last step is to create a "js" file and call it "slidein.js" and write a code in it.

```
$(function() {
    $(window).scroll(function(){
        var breakTop = $('#here').offset().top - $(window).height();
        if  ($(window).scrollTop() > breakTop)
            $('#slidein').animate({'right':'0px'},300);
        else
            $('#slidein').stop(true).animate({'right':'-430px'},100);
    });

    /* closing slidein */
    $('#slidein.close').bind('click',function(){
        $(this).parent().remove();
    });
});

```

Figure-125.Source code 78

The idea is that, in the text or article we have a paragraph with id=”here”. When we scroll pass it this module activates.

Finally, here is how it will look like:

Figure- 126.Example image 36

This box will slide out from the right bottom corner when we scroll pass the paragraph with id=”here”.

4.4 Slide up menu

Slide up menu also can be referred to as speed navigation (speed navi). Speed navi is a nice tool that will help users to access the most important site menu faster and easier.

Usually on websites there are 4-6 picture boxes that reflect the idea of menu, and when we click on them we are redirected to the needed place.

However, I decided to make it look more alluring using jQuery tool. The idea of the module developed by me was a slide up interaction. We see the text and when we hoover over it, text goes down and a picture appears. This all happens smooth and nice.

To do that, first of all we need to create a “default.php” file. Then we need to create “js” file and call it “speednavi.js”. Also we need to download a “js” file called “jquery.easing.1.3.js”. We can download it here:

https://www.dropbox.com/s/5tgvdy93kc2yk87/jquery.easing.1.3.js

Next we need to modify ”default.php” file to look like this:

```
<?php if (JRequest::getVar('view') == 'featured'): ?>
 <link rel="stylesheet" href="modules/mod_speednavi/css/style.css" type="text/css" media="screen"/>
</head>
<ul id="speednavi_box" class="speednavi_box"><li>
<a href="<?php echo $url1;?>"> <img src="<?php echo $purl1;?>" alt=""/> <span class="speednavi_active"></span> <span class="speednavi_wrap">
 <span class="speednavi_link"><?php echo $title1;?></span>
</span></a></li><li>
<a href="<?php echo $url2;?>"><img src="<?php echo $purl2;?>" alt=""/><span class="speednavi_wrap"><span class="speednavi_link"><?php echo $title2;?></span>
</span></a></li><li>
<a href="<?php echo $url3;?>"><img src="<?php echo $purl3;?>" alt=""/><span class="speednavi_wrap"><span class="speednavi_link"><?php echo $title3;?></span>
</span></a></li><li>
<a href="<?php echo $url4;?>"><img src="<?php echo $purl4;?>" alt=""/><span class="speednavi_wrap"><span class="speednavi_link"><?php echo $title4;?></span>
</span></a></li></ul>
 <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.min.js"></script>
 <script type="text/javascript" src="js/jquery.easing.1.3.js"></script>
 <script src="modules/mod_speednavi/js/speednavi.js"></script>
 <?php endif?>
```

Figure-127.Source code 79

The code above is not sorted as it should be, because it would not fit the screen otherwise.

That code has a list with four headings. Those headings have “urls”. Also, they have spans that make different parts of module look different.

All those elements will be modified by “css” later.

Next step is to write our code in “js” into “speednavi.js”.

```
$(function() {
        $('#speednavi_box > li').bind('mouseenter',function(){
            var $item = $(this);
            $item.find('img')
                 .stop(true)
                 .animate({
                    'width':'237px',
                    'height':'150px',
                    'left':'0px'
                 },400,'easeOutBack').andSelf().find('.speednavi_wrap').stop(true).animate({'top':'75px'},500,'easeOutBack').andSelf()
                 .find('.speednavi_active')
                 .stop(true)
                 .animate({'height':'170px'},300,function(){
                var $open_menu = $item.find('.sdt_box');
                if($open_menu.length){
                    var left = '170px';
                    if($item.parent().children().length == $item.index()+1)
                        left = '-170px';
                    $open_menu.show().animate({'left':left},200);}});
        }).bind('mouseleave',function(){
            var $item = $(this);
            var $open_menu = $item.find('.sdt_box');
            if($open_menu.length)
                $open_menu.hide().css('left','0px');
            $item.find('.speednavi_active')
                 .stop(true).animate({'height':'0px'},300).andSelf().find('img').stop(true).animate({
                    'width':'0px',
                    'height':'0px',
                    'left':'85px'},400).andSelf()
                 .find('.speednavi_wrap')
                 .stop(true)
                 .animate({'top':'25px'},500);});});
```

Figure-128.Source code 80

This code mainly means, that when we hover over the text, it slides down and instead of it appears a picture with link.

The last step is “css” code for “style.css” file. The code will be placed into appendices.

All together it works like this:

Figure- 129.Example image 37

Figure- 130.Example image 38

4.5 Drop Down menu

This is the last module I have developed. However, this module does not work with jQuery. This module gets its effects from "css". Using this module we will compare how jQuery differs from simple plugins.

This module works the same way as others, but it does not have those smooth interactions as jQuery modules.

To start with it, first of all we need to create a "default.php" file.

Then we modify it to look as below:

1st part	2nd part
<link href="modules/mod_dropdown_menu/style.css" rel="stylesheet" type="text/css">	<a href="<?php echo $urlParent3;?>"><?php echo $Parent3;?>
<ul id="nav">	
<a href="<?php echo $urlParent1;?>"><?php echo $Parent1;?>	<a href="<?php echo $urlChild15;?>"><?php echo $Child15;?>
	<a href="<?php echo $urlChild16;?>"><?php echo $Child16;?>
<a href="<?php echo $urlChild1;?>"><?php echo $Child1;?>	
<a href="<?php echo $urlParent5;?>"><?php echo $Parent5;?>	
	<a href="<?php echo $urlParent4;?>"><?php echo $Parent4;?>
<a href="<?php echo $urlChild2;?>"><?php echo $Child2;?>	
<a href="<?php echo $urlChild3;?>"><?php echo $Child3;?>	<a href="<?php echo $urlChild17;?>"><?php echo $Child17;?>
<a href="<?php echo $urlChild4;?>"><?php echo $Child4;?>	<a href="<?php echo $urlChild18;?>"><?php echo $Child18;?>
	<a href="<?php echo $urlChild19;?>"><?php echo $Child19;?>
	<a href="<?php echo $urlChild20;?>"><?php echo $Child20;?>
	<a href="<?php echo $urlChild21;?>"><?php echo $Child21;?>

```
<li><a href="<?php echo $urlChild5;?>"><?php echo $Child5;?></a></li>
<li><a href="<?php echo $urlChild6;?>"><?php echo $Child6;?></a></li>
<li><a href="<?php echo $urlChild7;?>"><?php echo $Child7;?></a></li>
</ul>
</li>
<li><a href="<?php echo $urlParent2;?>"><?php echo $Parent2;?></a>
<ul>
<li><a href="<?php echo $urlChild8;?>"><?php echo $Child8;?></a></li>
<li><a href="<?php echo $urlChild9;?>"><?php echo $Child9;?></a></li>
<li><a href="<?php echo $urlChild10;?>"><?php echo
$Child10;?></a></li>
<li><a href="<?php echo $urlChild25;?>"><?php echo
$Child25;?></a></li>
<li><a href="<?php echo $urlParent6;?>"><?php echo $Parent7;?></a>
<ul>
<li><a href="<?php echo $urlChild11;?>"><?php echo
$Child11;?></a></li>
<li><a href="<?php echo $urlChild12;?>"><?php echo
$Child12;?></a></li>
<li><a href="<?php echo $urlChild13;?>"><?php echo
$Child13;?></a></li>
<li><a href="<?php echo $urlChild14;?>"><?php echo
$Child14;?></a></li>
</ul></li></ul></li>
```

```
<li><a href="<?php echo $urlParent6;?>"><?php echo $Parent6;?></a>
<ul>
<li><a href="<?php echo $urlChild22;?>"><?php echo $Child22;?></a></li>
<li><a href="<?php echo $urlChild23;?>"><?php echo $Child23;?></a></li>
</ul>
</li>
<li><a href="<?php echo $urlParent7;?>"><?php echo $Parent7;?></a>
<ul>
<li><a href="<?php echo $urlChild24;?>"><?php echo $Child24;?></a></li>
</ul>
</li>
</ul>
</li>
```

Table-2

In this code we just created several lists as we want our menu to look like.

Next step is to modify its outlook and appearance using “css”.

We should write a “css” code into “style.css” file as the code in appendices.

After all this, the plugin will look like bellow:

Figure- 131.Example image 39

4.6 Conversion to Joomla compatible plugin

In order use plugins that we develop with Joomla CMS we need to convert them to compatible state. We cannot just utilize jQuery or other plugins in Joomla.

First of all, we need to keep in mind the fact that Joomla plugin is always a “zip” file. This means that we need to send a plugin folder to a “zip” state after we complete it.

To build a plugin, first of all we need to build its hierarchy. First of all we have a root folder of a plugin which name has “mod_” in the beginning. For example if my plugin’s name is “image rotator”, the plugin folder will be called something like “mod_image_rotator”.

Then inside this folder we create files and folders listed below:

- -css(folder)
- ----style.css(file)
- -tmpl(folder)
- ----default.php(file)
- ----index.html(file)
- -js(folder)
- ----jquery.js(file)
- -helper.php(file)
- -index.html(file)
- -mod_MODULE_NAME.php(file)
- -mod_MODULE_NAME.xml(file)

Some of these files and folders are already familiar for us, others are not.

Let’s get a closer look at the hierarchy and contents.

As we already know in “css” folder we will have “style.css” file for changing appearance.

“Js” folder has a jQuery files that will be used later by modules.

“Tmpl” folder has the most valuable information: “default.php” file. “Default.php” file is the main file where our main code is located.

“Default.php” file content should look like this:

```php
<?php
/**
 * @package     Joomla.Site
 * @subpackage  mod_userdata
 * @copyright   Copyright (C) 2005 - 2012 Open Source Matters, Inc. All rights reserved.
 * @license     GNU General Public License version 2 or later; see LICENSE.txt
 */

// no direct access
defined('_JEXEC') or die; ?>

<div class="moduletable<?php echo $params->get( 'moduleclass_sfx' ) ?>">

<ul>
<?php for ($i=0;$i< sizeof($list["users"]); $i++) { ?>

<li>
<?php if ($params->get( 'user_id' )) { ?>
<span><?php echo $list["users"][$i]["id"];?></span>
<?php } ?>
<?php if ($params->get( 'user_name' )) { ?>
<span><?php echo $list["users"][$i]["name"];?></span>
<?php } ?>
<?php if ($params->get( 'user_username' )) { ?>
<span><?php echo $list["users"][$i]["username"];?></span>
<?php } ?>
</li>

<?php } ?>
</ul>

</div>
```

Figure-132.Source code 81

We can leave “index.html” in the “tmpl” folder empty.

Next file is “helper.php” which should look like this:

```php
<?php
/**
 * @package     Joomla.Site
 * @subpackage  mod_userdata
 * @copyright   Copyright (C) 2005 - 2011 Open Source Matters, Inc. All rights reserved.
 * @license     GNU General Public License version 2 or later; see LICENSE.txt
 */ // no direct access
defined('_JEXEC') or die;
 class modUserDataHelper{
function getData( &$params ){
// Database query
$list = array();
$query = " SELECT id, name, username "
." FROM #__users "
." WHERE block=0 "
." ORDER BY id DESC "
." LIMIT " . $params->get( 'limit' );
$db =& JFactory::getDBO();
$db->setQuery( $query );
$rows = $db->loadObjectList();

// Get list items
if ($rows!=null)
{
$i=0;
foreach ($rows as $row)
{
$list["users"][$i]["id"]=$row->id;
$list["users"][$i]["name"]=$row->name;
$list["users"][$i]["username"]=$row->username;
$i++;
}return $list;}}}
```

Figure-133.Source code 82

“Index.html” file in root folder should also be empty.

Content of “mod_MODULE_NAME.php” file should be as below:

```php
<?php
/**
 * @package Joomla.Site
 * @subpackage  mod_userdata
 * @copyright   Copyright (C) 2005 - 2011 Open Source Matters, Inc. All rights reserved.
 * @license GNU General Public License version 2 or later; see LICENSE.txt
 */

// no direct access
defined('_JEXEC') or die;

// Include the syndicate functions only once
require_once dirname(__FILE__).'/helper.php';



// Get the layout
require JModuleHelper::getLayoutPath('mod_', $params->get('layout', 'default'));
```

Figure-134.Source code 83

Finally, content of “mod_ MODULE_NAME.xml” is shown below:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<extension type="module" version="1.7" client="site" method="upgrade">
<name>Demo</name><author>Izzat Nadiri</author><creationDate>21/11/2011</creationDate><copyright>Copyright (C) 2011. All rights reserved.</copyright>
<license>http://www.gnu.org/licenses/gpl-2.0.html GNU/GPL</license>
<authorEmail>izzatnadiri@gmail.com</authorEmail><authorUrl>google.fi</authorUrl><version>1.7.1</version>
<description>Demo</description><languages></languages>
<files>
<filename module="mod_">mod_.php</filename>
<filename>mod_.xml</filename>
<filename>helper.php</filename>
<filename>index.html</filename>
<folder>tmpl</folder>
</files>
<config>
<fields name="params">
<fieldset name="basic">
<field name="moduleclass_sfx" type="text" default="" label="Module Class Suffix" description="Suffix for individual css styling" />
<field name="limit" type="text" default="10" label="Limit Displayed Users" description="Limit Displayed Users" />
<field name="user_id" type="radio" default="1" label="Display user ID" description="Display user ID">
<option value="0">JNO</option>
<option value="1">JYES</option>
</field>
<field name="user_name" type="radio" default="1" label="Display Name" description="Display Name">
<option value="0">JNO</option>
<option value="1">JYES</option>
</field>
<field name="user_username" type="radio" default="1" label="Display Username" description="Display Username">
<option value="0">JNO</option>
<option value="1">JYES</option>
</field>
</fieldset></fields></config>
</extension>
```

Figure-135.Source code 84

Main files in this hierarchy are “mod_ MODULE_NAME.xml” and “default.php”. Always pay close attention to “mod_ MODULE_NAME.xml” file. List the names of included files right, because this file sets up the content of the module.

```xml
<extension type="module" version="1.7" client="site" method="upgrade">
<name>Demo</name>
<author>Izzat Nadiri</author>
<creationDate>21/11/2011</creationDate>
<copyright>Copyright (C) 2011. All rights reserved.</copyright>
<license>http://www.gnu.org/licenses/gpl-2.0.html GNU/GPL</license>
<authorEmail>izzatnadiri@gmail.com</authorEmail>
<authorUrl>google.fi</authorUrl>
<version>1.7.1</version>
<description>Demo</description>
<languages>
</languages>
<files>
```

```
<filename module="mod_">mod_.php</filename>
<filename>mod_.xml</filename>
<filename>helper.php</filename>
<filename>index.html</filename>
<folder>tmpl</folder>
</files>
```

Table-3

First line code means the version of Joomla, with which this module will be compatible. Joomla has number of versions such as 1, 1.5, 1.6, 1.7, 2.0, 2.5 etc. Second line is the name of module. Next line is the author's name. Then comes the date of customization. Next is copyright, which is very important. Without a copyright, your rights on the module will be violated. Then is the email of an author. The author's url refers to, the link to the webpage of the author. Then comes version, this time it is related to the module. Description can include some instruction or guideline. Last and the most important part are files. There we include the files to module. If we will not type the names of all files we include, later on our plugin will not be able to recognize them.

After we have completed all the steps, we can send the root folder to "zip" and keep it as a sample plugin. Then we can copy it and modify, and get a new module.

Joomla is an open source and is being developed every day. Joomla does not have all the plugins that are met on the internet nowadays, that is why it is very common when independent users develop their own plugins. Later on it is possible to sell plugins or upload for free to Joomla website.

5 CONCLUSION

The purpose of this project was developing a modern website for company MHG Systems Oy. The main idea was to create eye catching webpage without using flash or other heavy methods that will make the site work slowly or load long.

Secondly, the goal was to convert customized plugins into Joomla CMS, because the company was originally using Joomla for website development.

The goal was achieved; a number of plugins were customized and converted later on to Joomla.

To start with this project, we had to get some knowledge in several programming languages and tools such as HTML, CSS, PHP, JavaScript, jQuery etc.

As our main topic was development of jQuery plugins, we focused mainly on learning it. The second chapter is almost fully dedicated to jQuery plugin development tutorials.

However, other parts of the project were not left untouched. We also discussed installation of Joomla, module development, important tools for usage etc.

When I started this project I had no experience with jQuery. I only had experience in PHP, CSS, HTML and to some extent in Joomla. I was learning while doing.

I started from customizing already existing plugins, and later on I decided to develop my own modules. The reason for that was, that modules I was asked to add to the site did not exist in compatibility with Joomla. That is why I decided to create them myself.

The reason why I have done this project using jQuery is that, it allows us to do the same interactions without any lags or latency in loading. Personally I prefer jQuery over Flash. In my opinion, Flash works much slower than jQuery.

Firstly, this project was not intended to be as wide as it is now. I was originally planning to develop fixed modules which can be utilized only by MHG Systems Oy. However, later I understood that, I can extend this project and utilize plugins later on.

All together I developed four plugins:

- Language switcher
- Slide-in box
- Slide-up menu
- Drop-down menu

All of them are very useful and interesting. They are not very hard to implement though.

Firstly, I decided to customize a plugin called “Language Switcher” which is in my opinion very widely used nowadays.

I found language selection methods of webpages nowadays very dull and boring that is why I decided to make it look fancier and smoother.

The idea of this module is quite simple. I customized it using a combination of jQuery, CSS, PHP etc.

In PHP I created a list which I styled with CSS later on. And the final stroke was jQuery which gave it interaction and smooth motions.

Final result is a nice looking drop down language switcher.

Next plugin was “slide-in box”. I was asked by the Business software developer of MHG System Oy. to develop this kind of plugin. That was a reason why I decided to customize it.

Also after a small research I realized that, Joomla does not have ready “slide-in box” kind of module.

Yet again, the idea of this plugin was a box sliding in the page after scrolling to a certain part of an article. This plugin is very common nowadays all over the internet.

I again combined PHP, CSS and jQuery in this module. First of all I created a “div” and styled it to look like a box, and added some text and links to it. Last step was jQuery which slides this box in, when the scroll passes a paragraph with an id=”here”. Apparently, we also have to set an id to a paragraph beforehand.

Next plugin is “slide-up menu” also referred to as “speed navigation”. This kind of module is common. We can see it on sites of great corporations such as IBM, Coca-Cola, Nike etc.

The main idea in this module is its smooth and attractive interaction.

Substantially, we have menus which are dropping down when hovered over, and instead a picture appears. When clicked that picture redirects us to the corresponding url.

This module as all others was developed using PHP, CSS and jQuery.

I started creating a list of four headings. Later I styled them in CSS to look good i.e. changed font-style, size, color etc. Last step again was jQuery, which added a nice touch to the plugin. After applying jQuery the text got a smooth interaction.

Last module is “drop down menu” which also was developed by me. However, the reason why I included it to this project is that it shows us clearly the difference between jQuery interactions and other interactions. This module is developed using only CSS and PHP. And when we compare it to other modules which had jQuery in them, we can see how crucially they differ. The main difference is interaction i.e. how the module moves. Drop down menu works very fast while jQuery modules work smooth and slow.

Last step was a conversion of plugins into Joomla compatible plugins.

BIBLIOGRAPHY

Rahmel Dan 2009, Beginning Joomla!, Second edition, Dan Rahmel, New York

Bellamy Seamus 2011, Joomla! For Dummies, Second edition, Wiley Publishing, Inc., Indianapolis

Wellman Dan 2009, jQuery UI 1.6: The User Interface Library for jQuery, First edition, Packt Publishing, Birmingham

Bibeault Bear, Katz Yehuda 2008, jQuery in Action, First edition, Manning Publications Co., Greenwich

Chaffer Jonathan, Swedberg Karl 2009, Learning jQuery 1.3, First edition, Packt Publishing, Birmingham

Benedetti Ryan, Cranley Ronan 2011, Head First jQuery, First edition, O'Reilly Media, Inc., Sebastopol

Sawyer McFarland David 2011, JavaScript & jQuery: The Missing Manual, Second edition, O'Reilly Media, Inc., Sebastopol

Bibeault Bear, Katz Yehuda 2010, jQuery in Action, Second edition, Manning Publications Co., Greenwich

Sharp Jonathan, Burns Rob 2010, jQuery Cookbook, First edition, O'Reilly Media, Inc., Sebastopol

Naramore Elizabeth, Gerner Jason, Le Scouarnec Yann, Stolz Jeremy, K. Glass Michael 2005, Beginning PHP5, Apache, and MySQL® Web Development, First edition, Wiley Publishing, Inc., Indianapolis

Duckett Jon 2010, Beginning HTML, XHTML, CSS, and JavaScript, First edition, Wiley Publishing, Inc., Indianapolis

Electronic sources

HTML Tutorial by w3schools [referred 21.07.2012]
Available in www-format:
<URL: http://www.w3schools.com/html/default.asp />.

CSS Tutorial by w3schools [referred 24.06.2012]
Available in www-format:
<URL: http://www.w3schools.com/css/ />.

PHP Tutorial by w3schools [referred 12.08.2012]
Available in www-format:
<URL: http://www.w3schools.com/php/default.asp />.

jQuery Tutorial by w3schools [referred 17.05.2012]
Available in www-format:
<URL: http://www.w3schools.com/jquery/default.asp/>.

JavaScript Tutorial by w3schools [referred 14.06.2012]
Available in www-format:
<URL: http://www.w3schools.com/js/default.asp/>.

jQuery vs Flash Article by Justin Simonelli. [referred 9.12.2012]
Available in www-format:
<URL: http://www.insivia.com/is-jquery-taking-over-flash/>

Introduction to jQuery by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=1>.

Implementing jQuery by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=3>.

Testing jQuery by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=4>.

Ready by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=7>.

Load by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=8>.

Introduction to Selectors by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=11>.

All selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=12>.

ID selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=13>.

Element selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=14>.

Submit selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=15>.

Text selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=16>.

Multiple selectors by Alex Garrett [referred 9.03.2012]
Available in video-format:

<URL: http://thenewboston.org/watch.php?cat=32&number=18>.

This selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=19>.

Even/odd selectors by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=20>.

Attribute selectors by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=21>.

Contains selector by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=23>.

Font size switcher by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=27>.

Hover by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=37>.

Scroll by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=38>.

Character counter by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=45>.

Val by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=53>.

Fade in by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=68>.

Fade toggle by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=70>.

Slide down by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=71>.

Stop by Alex Garrett [referred 9.03.2012]

Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=74>.

Delay by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=75>.

Gallery fading effect by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=76>.

Scroll to top by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=89>.

Enable a checkbox after scroll by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=90>.

Show password by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=116>.

Learning jQuery UI (User Interface) by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=139>.

Installing jQuery UI (User Interface) by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=140>.

Draggable by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=141>.

Droppable by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=141>.

Sortable by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=147>.

Resizable by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=149>.

Accordion by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=152>.

Datepicker by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=154>.

Progress bar by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=160>.

Slider by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=161>.

Tabs by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=163>.

Drag and drop list by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=166>.

Sliding down message by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=171>.

Emoticons by Alex Garrett [referred 9.03.2012]
Available in video-format:
<URL: http://thenewboston.org/watch.php?cat=32&number=192>.

APPENDIX/APPENDICES
Appendix 1 Language switcher style.css file

```
#country-select {
position: absolute;
top: 13px;
right: 0;
width: 180px;
}
#country-select form {
width: 180px;
padding: 0;
}
#country-select select,
#country-select input {
display: inline;
padding: 0;
margin: 0;
}
.dropdown dd { position: relative; }
.dropdown a {
text-decoration: none;
outline: 0;
font: 12px Arial, Helvetica, sans-serif;
display: block;
width: 150px;
overflow: hidden;
}
/*color*/.dropdown dt a {
background: transparent;
border: 1px solid #;
padding: 3px 10px 4px 10px;
color:#FF9933 ;
}
.dropdown dt a.active {
width:123px;
background: transparent ;
-webkit-border-bottom-right-radius:     0;-webkit-border-bottom-left-
radius: 0;
-moz-border-radius-bottomleft: 0;
-moz-border-radius-bottomright: 0;
```

```
border-bottom-left-radius: 0;
border-bottom-right-radius: 0;
border-bottom: 1px dotted #676768;
-moz-box-shadow: 0 3px 7px rgba(0,0,0,.5);
-webkit-box-shadow: 0 3px 7px rgba(0,0,0,.5);
box-shadow: 0 3px 7px rgba(0,0,0,.5);
color: #FF9933 ;
}
.dropdown dd ul {
width:142px;
background-
im-
age:url('/mhgsystems.com/templates/beez5/images/LayoutIcons/language_
background.png');
border: 1px solid #676768;
color: #C5C0B0;
display: none;
position: absolute;
z-index: 999;
top: 0;
left: 0;
padding: 2px 0 5px 0;
list-style: none;
border-top: none;
margin: 0;
-moz-box-shadow: 0 3px 7px rgba(0,0,0,.5);
-webkit-box-shadow: 0 3px 7px rgba(0,0,0,.5);
box-shadow: 0 3px 7px rgba(0,0,0,.5);
}
.dropdown dd ul :hover{
background:#669933 ;
width:auto;
}
.dropdown dd ul li a {
padding: 2px 10px;
}
.dropdown dd ul li a span,
.dropdown dt a span {
float: left;
width: 16px;
height: 11px;
margin: 2px 6px 0 0;
```

```
background-image: url(flags.png);
background-repeat: no-repeat;
cursor: pointer;
}
.dropdown dd ul li a em,
.dropdown dt a em {
font-style: normal;
float: left;
width: 100px;
cursor: pointer;
}
.dropdown dd ul li a em {
color: #FF9933 ;
}
.dropdown dd ul li a:hover { }
.dropdown dd ul li a:hover em { color:#FF9933 ; width:100px;}
```

Appendix 2 Language switcher .js file

```
$(document).ready(function() {
openList();
var $openList = $(".dropdown dt a");
var $dropItems = $(".dropdown dd ul");
$openList.toggle(function() {
$dropItems.slideDown(200);
$openList.addClass("active");
}, function() {
$dropItems.slideUp(200);
$(this).removeAttr("class");});
$(document).bind('click', function(e) {
var $selected = $(e.needed);
if (! $selected.parents().hasClass("dropdown"))
$dropItems.slideUp(200);
$openList.removeAttr("class");
});
$(".dropdown dd ul li a").click(function() {
var chosenItem = $(this).parent().attr("class");
var chosenTitle = $(this).find("em").html();
$("#needed dt").removeClass().addClass(chosenItem);
$("#needed dt em").html(chosenTitle);
$dropItems.hide();
$openList.removeAttr("class");
```

```
 });});
 function openList(){
 var $form = $("div#country-select form");
 $form.hide();
 var source = $("#country-options");
 source.removeAttr("autocomplete");
 var selected = source.find("option:selected");
 var options = $("option", source);
 $("#country-select").append('<dl                          id="needed"
 class="dropdown"></dl>')
 $("#needed").append('<dt   class="'   +   selected.val()   +   '"><a
 href="#"><span   class="flag"></span><em>'   +   selected.text()   +
 '</em></a></dt>')
 $("#needed").append('<dd><ul></ul></dd>')
 options.each(function(){
 $("#needed  dd  ul").append('<li  class="'  +  $(this).val()  +  '"><a
 href="'       +       $(this).attr("title")       +       '"><span
 class="flag"></span><em>' + $(this).text() + '</em></a></li>');
 });
}
```

Appendix 3 Slide-up menu style.css file

```
ul.speednavi_box{
margin:0;
padding:0;
list-style: none;
font-family:Arial,'DejaVu   Sans','Liberation   Sans',Freesans,sans-
serif;
font-size:10px;
width:1000px;}
ul.speednavi_box a{
text-decoration:none;
outline:none;}
ul.speednavi_box li{
float:left;
width:158px;
height:56px;
position:relative;
cursor:pointer;}
/*main box */
ul.speednavi_box li > a{
```

```
/*background-
im-
age:url('http://localhost/latest/templates/beez5/images/LayoutIcons/s
peed_navi.png');*/
background:transparent;
border:transparent;
position:absolute;
top:0px;
left:0px;
width:158px;
height:100px;
z-index:12;}
ul.speednavi_box li a img{
border:none;
position:absolute;
width:0px;
height:0px;
bottom:0px;
left:85px;
z-index:100;}
ul.speednavi_box li span.speednavi_wrap{
position:absolute;
top:25px;
left:0px;
width:158px;
height:60px;
z-index:15;}
ul.speednavi_box li span.speednavi_active{
position:absolute;
background:transparent;
top:85px;
width:240px;
height:0px;
left:0px;
z-index:14;}
ul.speednavi_box li span span.speednavi_link{
position:relative;
left:37px;
color:grey ;
font-size:20px;}
ul.speednavi_box li span span.sdt_descr{
color:#0B75AF;
```

```
float:left;
clear:both;
width:158px;
font-size:10px;
letter-spacing:1px;}
l.speednavi_box li div.speed_square{
display:block;
position:absolute;
width:158px;
overflow:hidden;
height:170px;
top:85px;
left:0px;
display:none;
background:#000;}
ul.speednavi_box li div.speed_square a{
float:left;
clear:both;
line-height:30px;
color:#0B75AF;}
ul.speednavi_box li div.speed_square a:first-child{
margin-top:15px;}
ul.speednavi_box li div.speed_square a:hover{
color:#fff;}
```

Appendix 4 Drop down menu style.css file

```
a {
color: #333;
}
#nav {
margin: 0;
padding: 7px 6px 0;
line-height: 120%;
}
#nav li {
margin: 0 5px;
padding: 0 0 8px;
float: left;
position: relative;
list-style: none;
}
/* main level link */
```

```
#nav a {
font-size:17px;
font-weight: bold;
color: #e7e5e5;
text-decoration: none;
display: block;
padding:  8px 20px;
margin: 0;
-webkit-border-radius: 1.6em;
-moz-border-radius: 1.6em;
text-shadow: 0 1px 1px rgba(0,0,0, .3);
}
#nav a:hover {
/*
background: #000;
color: #fff;
*/
}
/* main level link hover */
#nav .current a{
/*
background:     #666     url(http://www.webdesignerwall.com/demo/css3-
dropdown-menu/img/gradient.png) repeat-x 0 -40px;
color: #444;
height:35px;
width:193px;
-webkit-box-shadow: 0 1px 1px rgba(0,0,0, .2);
-moz-box-shadow: 0 1px 1px rgba(0,0,0, .2);
box-shadow: 0 1px 1px rgba(0,0,0, .2);
text-shadow: 0 1px 0 rgba(255,255,255, 1);
*/
}
#nav li:hover > a {
background:     #666     url(http://www.webdesignerwall.com/demo/css3-
dropdown-menu/img/gradient.png) repeat-x 0 -40px;
color: #444;
-webkit-box-shadow: 0 1px 1px rgba(0,0,0, .2);
-moz-box-shadow: 0 1px 1px rgba(0,0,0, .2);
box-shadow: 0 1px 1px rgba(0,0,0, .2);
text-shadow: 0 1px 0 rgba(255,255,255, 1);
}
/* sub levels link hover */
```

```
#nav ul li:hover a, #nav li:hover li a {
background: none;
border: none;
color: #666;
width:180%;
-webkit-box-shadow: none;
-moz-box-shadow: none;
}
#nav ul a:hover {
background: #00000;
color: #000 !important;
-webkit-border-radius: 0;
-moz-border-radius: 0;
text-shadow: 0 1px 1px rgba(0,0,0, .1);
}
/* dropdown */
#nav li:hover > ul {
width:200px;
display: inline;
z-index:10000000;
}
/* level 2 list */
#nav ul {
display: none;
margin: 0;
padding: 0;
position: absolute;
top: 24px;
left: 0px;
background: #ddd url(http://www.webdesignerwall.com/demo/css3-
dropdown-menu/img/gradient.png) repeat-x 0 0;
border: solid 1px #b4b4b4;
-webkit-border-radius: 10px;
-moz-border-radius: 10px;
border-radius: 10px;
-webkit-box-shadow: 0 1px 3px rgba(0,0,0, .3);
-moz-box-shadow: 0 1px 3px rgba(0,0,0, .3);
box-shadow: 0 1px 3px rgba(0,0,0, .3);
}
#nav ul li {
float: none;
width:100%;
```

```
margin: 0;
padding: 0;
}

#nav ul a {
font-weight: normal;
text-shadow: 0 1px 0 #fff;
}
/* level 3+ list */
#nav ul ul {
position:absolute;
left:80%;
top: 19px;
}
/* rounded corners of first and last link */
#nav ul li:first-child > a {
-webkit-border-top-left-radius: 9px;
-moz-border-radius-topleft: 9px;
-webkit-border-top-right-radius: 9px;
-moz-border-radius-topright: 9px;
}
#nav ul li:last-child > a {
-webkit-border-bottom-left-radius: 9px;
-moz-border-radius-bottomleft: 9px;
-webkit-border-bottom-right-radius: 9px;
-moz-border-radius-bottomright: 9px;
}
#nav:after {
content: ".";
display: block;
clear: both;
visibility: hidden;
line-height: 0;
height: 0;
}
#nav {
display: inline-block;
}
```