

Metropolia University of Applied Sciences
Institute of Technology
Degree Programme in Media Engineering

Tadhg Clancy

**Re-Designing a Company Website: Creating a Website
Administration**

Bachelor's Thesis: 19 November 2009

Instructor: Inka Niteplöd, Liikkunen Owner
Supervisor: Harri Airaksinen, Lecturer

Abstract

Author	Tadhg Clancy
Title	Re-designing a company website: creating a website administration
Number of Pages	44
Date	19 November 2009
Degree Program	Media Engineering
Instructor	Inka Niteplöd, Liikkunen Owner
Supervisor	Harri Airaksinen, Lecturer
<p>The goal of the final year project was to redesign a website Liikkunen.fi for the company Liikkunen, so that the owner could maintain the site. The existing site was developed with VBScript, therefore for updates to precede one needed to have knowledge of VBScript.</p> <p>The solution was a redesigned, PHP and MySQL based site with an administration area. The site was completely redesigned so that the layout would be clearer and consequently the user could navigate more easily. Problems were encountered with regular expressions for names and also for converting the site into Unicode; both problems were solved so a satisfactory level where the site operates without hindrance. The administration area allows the owner to add, modify and delete the site content without having to search or pay for a third party's involvement. It also lists the company's customers and all their details in manageable lists. The main problem found was comparing Scandinavian letters in queries such as 'ä' to 'ä'. Only by converting the page format and the character encoding to UTF-8 was the desired outcome attained.</p> <p>Encoding the site from scratch may have been made easier by using a prebuilt system such as a content management system (CMS). However the site posed a challenge and the owner is satisfied with it. One change that could have been made would be to include the course date for the registration. This would enable sorting through the course name and course date. Furthermore it would have made it easier for the owner as there would not have to be a new course made to accommodate the second or third dates for a course and its registration. The site could also be further developed so that the owner could create their own links and develop new pages for their site.</p>	
Keywords	web design, site administration, PHP, MySQL

Contents

1 Introduction	4
2 Existing Site	5
2.1 Initial Reactions	5
2.2 Logo Placement	6
2.3 Links	7
2.4 Extra Features	7
3 Designing the User Interface	8
3.1 Design	8
3.2 The use of Object Oriented Programming PHP	12
3.3 MySQL and SQL queries	16
3.4 Cookies	19
3.5 Regular Expressions	20
3.6 Browser Compatibilities	21
3.7 Multilingual Sites	23
4 Designing the Administration Interface	25
4.1 Design	25
4.2 Admin Usage of OOP PHP	27
4.3 Security	30
4.4 Admin Site Multilingual Problems and Solutions	31
4.5 Adding – Modifying – Deleting Data	33
5 Conclusions	36
References	37
Appendix 1:	39
Further Examples of Linking Discrepancies	39
Appendix 2:	41
Various Code.	41

1 Introduction

Liikkunen, the company, was only bought January 2009; at the time there was an old site present created in ASP where the owner had to call the original builder to make any changes to the site. Inka Niteplõd, who is the new owner, wanted to have a new website where she could update the site without having to go to a third party. This in turn would also save the company money; there would be no more third party dealings. Liikkunen is a gymnastics company aimed at families; they deal with Expectant Mothers, Newborn's and Mothers, Toddlers and Parents, Children and finally Adults.

There were two choices of programming languages for this project, PHP, or XML with MySQL. PHP was chosen because Nebula, Liikkunen's web-host, was unable to host XML based sites, but was able to host PHP with MySQL. The site would allow the customer complete control of the site and all its features; she could create, delete and modify everything.

This thesis will cover the creation of the new site, from its first mock-ups to the final version. Section 2 will detail some of the main flaws that existed in the current site from the Logo to the links. Section 3 deals with the initial design, the use of PHP and MySQL with the problems occurring due to Regular Expressions, Cookies and Multilingual sites, included in this is the issues with Browser Compatibilities. Section 4 is the administration where Security and Multilingual Site problems occur. This will also detail how to add / modify / delete courses and other information to the site. Finally, in Section 5, there will be the conclusions where the site will be assessed and further improvements of changes could / can be made.

2 Existing Site

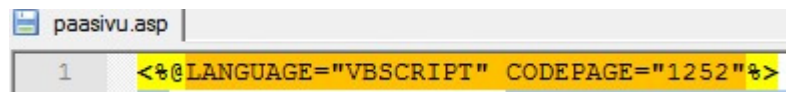
2.1 Initial Reactions

Upon first inspection of the Liikkunen website it was presumed that the site was created for children. Upon further inspection one realised that the site was for Mothers-to-be, new Mothers, etc. The point of the site was unknown to me, it looked like a child's site to learn different information, such as counting and colours, I had to think. The best method a site can use is giving its audience the point of the site straight away. At first glance without scanning or reading a site is supposed to give the user its meaning, if it does not then the site's purpose is lost. It makes you think [1]. Figure 1 is the front page of the site.



Figure 1. Liikkunen front page.

The site was originally written in VBScript which is a Microsoft scripting language [2]. This was discovered when the front page 'paasivu.asp' was opened up in Notepad++. Figure 2 shows the first line for every .asp page that was used in the site.



```
paasivu.asp
1 <%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
```

Figure 2. First line from 'paasivu.asp'

Due to the site being coded with VBScript, the site was locked into using an outside / third party to update the site. Also enforcing this issue was the fact that there was no site administration. Due to there being no administration, the owner had to call the original maker to insert / modify / change the text. This over time would cost the site owner valuable capital which could be used for other endeavours.

2.2 Logo Placement

Site Logo placement is one of the most effective methods to dictate to the user where they are; that they have not left the site. Unfortunately for the Liikkunen site, one can only see the logo on the front page, figure 1 demonstrates this. The other pages do not display the logo, appendix 1 figures 1 and 3 show this.

On top of having the logo seen on all pages the logo should be placed in a location that is both natural and fixed. For Europe and other Western countries, we read from left to right. Therefore, it would be common for the user to look at the top left of the site or header for the logo [3]. The logo on the Liikkunen paasivu is neither at the top nor in the header; it is located in the main text area of the site.

The site logo is like a building name. Once one enters a shop they know where they are until they leave, unlike the web. The web provides links to different parts of the site. One will not know they are still on the same site unless they see the site logo on every page that they visit [1]. If a site does not have a logo, one could be transported to another area of the Internet, the only way they know where they are would be the URL. If the URL remains constant, it means they are on the same site; if it changes, they know they have moved to a different area of the web.

2.3 Links

Links should be in a different colour or format from the normal text to indicate that they are links. Customers are left baffled as knowing to what is, or is not, clickable content if link are not clearly defined. There are many instances in the site where this is not adhered to. The general look of a link is that its colour is blue and / or underlined. It is the default colour unless changed. On the Liikkunen site the links look like normal text until they are highlighted.

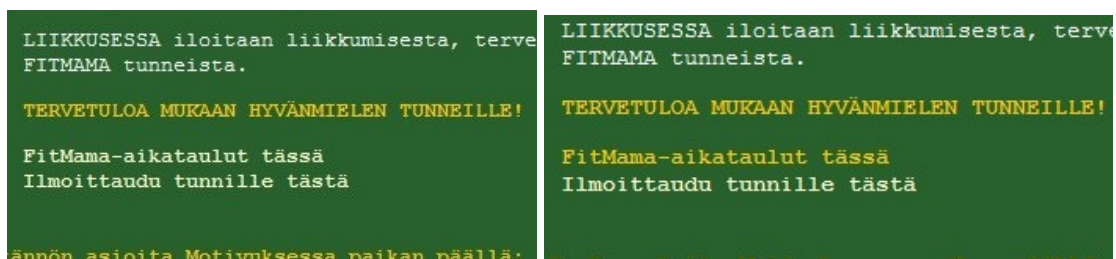


Figure 3. Unknown links

Figure 3 shows the difference in the linking system that is used in this site. One cannot tell straight away if it is a link or not until one hover's the mouse over it. Appendix 1 demonstrates more examples of this, including a full scale image of the current page (jumpat_mamma.asp). The best method for links would be to have them either another colour (preferably blue) and underlined [4]. On this site neither is done, links look like the normal text so that one does not know what is clickable or not.

2.4 Extra Features

A surprising feature of the site is when one hovers over the “chalk” at the bottom of the “black board”. The site background colour changes to the colour of the chalk. Also, when one covers over the “duster / sponge”, the background reverts back to the original colour. If one has a colour select from the chalk it does not stay for new pages, the background once again reverts to the original colour. If the site was to be consistent, the new background colour should have stayed for all pages.

A further and time consuming problem is when a new / existing client signs up for a course, the information is sent via email to the owners email. The owner then has to copy / paste all the client's information into an Excel file. This process can take hours every week for the owner to complete. A site administration would streamline this process.

3 Designing the User Interface

3.1 Design

During the first few meetings between the customer and I, we checked out the competitors' sites and also some sites that the customer was interested in or thought looked "nice". After a few weeks, we sketched a few mock-ups and then implemented these mock-ups in HTML and CSS. Figures 4 and 5 represent the basic ideas that we had both envisioned. Figure 5 is the closest mock-up to the actual site.

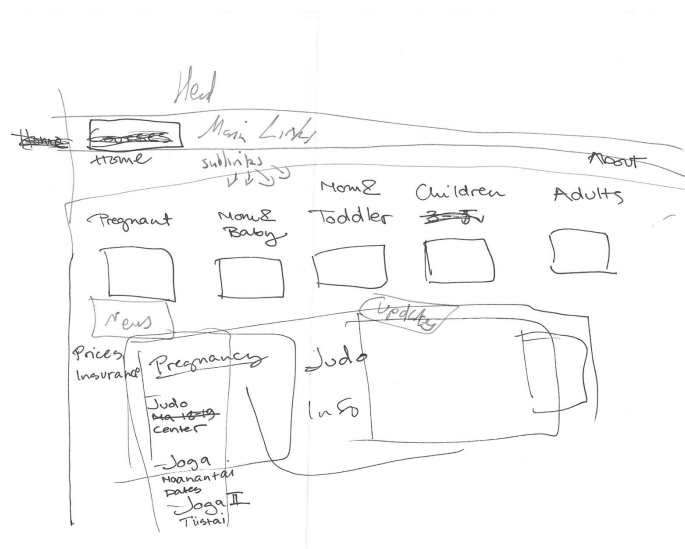


Figure 4. First Mock-up

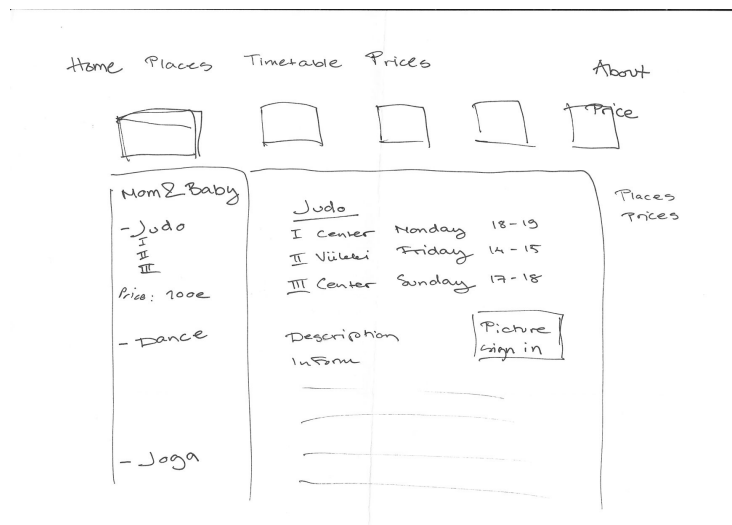


Figure 5. Second Mock-up

Websites have some basic needs, a container, header, main navigation, content and footer. The entire site should reside between these areas [3]. The container will hold everything; it also sets boundaries for the designers work. The 975px would be the container for the whole site. The header as stated in section 2.2 would reside in the top left hand corner and along with the header, the main navigation would be to the right of it. The main navigation could also be defined as persistent navigation where it remains almost always constant. It also gives confirmation to the user that they are still on the same site [1]. Since the site would only be 975px wide, having the links with the header was the most logical choice.

Next piece of navigation, secondary navigation, would be the course type links. Due to there possibly being tertiary navigation, and there is, having the links underneath the header was the only viable option. The tertiary links are the course links; these would change depending on what course type would be selected. As can be seen from figure 4, the basic of what has just been outlined is present.

Figure 5 is a much more refined version of the site; here the content area of the site is much more clearly defined. The content would hold all the course specific information. Content is not a design issue, but it does transcend design; no matter how good or bad looking a site is if there is nothing of value, content wise, to offer the user they will leave [4].

The overall width of the site was decided to be 975px. The reason being firstly, through my own experience of resolutions in both laptops and desktops the lowest resolution I have personally used was 1024*768. This on an old laptop which was used a number of years ago. Secondly the lowest resolution that is available in desktops is also 1024*768 [5]. Even amongst the gaming community the lowest resolution is 1024*768 [6]. As for laptops the only type of laptops that may not be able to view the entire width of the site are the mini laptops, Netbooks. Netbooks have smaller than normal screen sizes, such as 12.1” or smaller displays. These displays may only have 800*600 or fewer pixels [7]. Another reason for the site being less than 1000px is that the scroll bar in browsers will use up the remaining pixels that are left. If the right div was any larger, the right div would drop down below the images at the bottom of the site.

At first the customer wanted to keep the original colours since they already had some business / payment cards made and felt that it may be too expensive to renew the cards again. Figure 6 displays the original colours of the Liikkunen site and the business cards. But this was to change when a friend of the customer, a graphic designer, designed a new logo with new colours. He also created what was to be, in conjunction with our brainstorming, the general layout for the Liikkunen site. Figure 7 is the design for the new Liikkunen logo and the general look of the Liikkunen site.



Figure 6. Liikkunen Cards



Figure 7. Designers Mock-up

Figure 8 is the final design that is used in the Liikkunen site. It is a combination between Mock-ups 2 and 3; figure 5 and figure 7.

KOTI KURSSIT LUKUJÄRJESTYS

LUKUNURKKA GALLERIA LIIKUNTAPAIKAT

HINNASTO REKISTERÖIDY YHTEYSTIEDOT

MOTIVUS
smartum
etu käy meillä

Äiti ja vauva

Vauvajumppa 8 kk - 1,5 v [Ilmoittautuminen](#)

Leikkimielinen jumppatunti äidille ja vauvalle. Tunnilla yhdistetään äidin sykkeen kohottaminen ja lihasten vahvistaminen yhteiseen toimintaan vauvan kanssa.

Tuttujen "vauvajuttujen" lisäksi "isojen ryhmässä hypitään, pallotellaan ja ryömitään pikkuisella tempuradalla. Yhteistyö ja vauvan leikin osuus on suurempi kuin "pienen jumpassa".

Vauvajumppien ikäjako on viitteellinen. "Isojen" ryhmä on tarkoitettu vauhdikkaammille menijöille.

Kesto: 45min

Ajankohdat	Paikka	Päivä	Aika	Ohjaajat
31.08-7.12.2009	MOTIVUS Stockmannilla	Maanantai	14:30	Hanna-Maria Mäkelä

Site created by Tadhg Clancy

Koska olen tyttö.
LAUSETA YÄSTÄ

Yrittäjäkummi

Figure 8. Final Design, Kurssit.php.

Figure 8 has a header in the recommended position and it is also a link back to the homepage of the site [3]. The navigation is persistent throughout the site [1]. Also the navigation is clearly defined as it is in a different colour from the normal text and it is underlined when not a part of the main navigation [4]. The content area is defined with a lighter colour background so that the user can see the information clearly [3]. Although there is advertising on the site, it only accounts for approximately 25%, the rest is for the content [4].

3.2 The use of Object Oriented Programming PHP

PHP was chosen as the basis of my thesis as the language can easily manipulate database query results into the desired format. The PHP would be used to display all the required information needed for the site. It would also allow the customer to redefine the information once the site is complete.

Object Oriented Programming (OOP) in PHP first appeared in PHP 3 and was then improved in PHP 4 [8]. OOP PHP had a number of deficiencies in PHP 4 such as, an unorthodox object-referencing methodology, no standard convention for naming constructors and absence of object destructors [9]. In 2004 a new version of PHP was released PHP 5 and this version eliminated all the previous problems and added new OOP features.

There are different approaches to take with PHP when creating a website / web-application, one is a Procedural Approach and another is the Object Oriented approach. In a procedural approach you first have the Data Definitions and then the Manipulation Code. For the object oriented approach both the Data Definitions and the Manipulation Code are grouped together. Example:

Procedural Approach

- 1) Data Definitions
 - a. Data definitions for X
 - b. Data definitions for Y

- 2) Data manipulation code
 - a. Code for X
 - b. Code for Y

Object Oriented Approach

- 1) X
 - a. Data Definitions
 - b. Manipulation Code
- 2) Y
 - a. Data Definitions
 - b. Manipulation Code

Both versions will have their own positives and negatives and it is in the end up to the designer / code to decide which approach best suits them [10]. I chose the object oriented approach.

OOP PHP uses Classes to define and manipulate the code, for this Constructors and Methods are created. Code example 1 shows a Class.

```
<?php
class Form {
    private $formMethod;
    private $formAction;
    private $formEnctype;

    function __construct($nFormMethod, $nFormAction, $nFormEnctype){
        $this->formMethod=$nFormMethod;
        $this->formAction=$nFormAction;
        $this->formEnctype=$nFormEnctype;
    }

    public function getForm($content){
        $response = "<form action=\"".$this->formAction."\"
                    method=\"".$this->formMethod.\"\"
                    enctype=\"".$this->formEnctype.\"\">\n";
```

```
$response .= $content;
$response .= "</form>";
return $response;
}
}
?>
```

Code Example 1. Form Class.

For this Class, Form, has three private variables. The first function constructs the Form object and the public function displays the full class when called with:

```
$form = new Form($method, $action, $enctype);
echo $form->getForm($code);
```

Classes such as this can easily be called again and again without being edited. It can also reduce the work flow. Other Classes used in this project include Select, Input and MySQL, each have their own functions and can work together seamlessly.

[Sulje ikkuna](#)

Mammajumppa2

ILMOITTAUTUMINEN

Valitse ensin kurssi:

Vanhemman nimi:
Esim: Malla Mallikas

Laskettu aika:
Esim: 12-12-2000

Lähiosoite:
Esim: Kadunnimi 1 A 1

Postinumero:
Esim: 12345

Postitoimipaikka:
Esim: Paikka

Sähköpostiosoite:
Esim: nimi.nimi@yahoo.com

Puhelin:
Esim: 0501234567

Valitse maksu:

Sähköpostiini

Puhelimeeni

Maksa kurssimaksu Liikkusen pankkitilille. Tilinumero on Sampo-pankki 800011-71333405 / Liikkunen (löydät tilinumeron jatkossa kohdasta Yhteystiedot). Ota kuitti maksusta mukaan tunnille. Liikuntaseteleillä maksat suoraan ohjaajalle.

Kun tulet tunnille, Motivuksen tiskillä ei tarvitse ilmoittautua. Lippuasiat hoidetaan Liikkusen ohjaajan kanssa tunnin alussa.

Esteen sattuessa voit korvata kaksi kurssikertaa toisen vastaavan kurssin tunneilla tai vauvajumpassa. Korvauserroille ilmoittaudut sähköpostilla: asiakaspalvelu@liikkunen.fi.

[Sulje ikkuna](#)

Figure 9. Example Form.

Figure 9 represents how each of the Classes work together:

- 1) A query is sent to the database, the results depend on the ID that is sent via the URL. The mysql Class first displays the course name, in this case “Mammajumppa2”.
- 2) Another query is sent for the first select option, “Valitse ensin kurssi”. The mysql Class the returns a row(s). The row data is then used to populate the

options, which is created by the Select Class. The number of courses in the list is determined if they are open for registration.

- 3) The Input boxes are then created with the Abstract Input Class and its Extended Classes InputText or InputTextArea. Depending on the “course type” will determine the Input that the user must fill.
- 4) The second select option “Valitse maksu” is created using the same method as the first drop down list
- 5) Next the payment information is displayed. First a query is sent to the database, second the result is then processed by the mysql Class, and lastly the result from the mysql Class can then be displayed on the page. This result is dependent on the course. Different courses have different payment information.
- 6) Finally the Form Class is called and all the resultant code is then embedded in a Form.

3.3 MySQL and SQL queries

MySQL is an open-source relational database. Static HTML pages provide no viable means of instant updates; one can only open the file and recode the new information. MySQL along with PHP provide that answer; using both together one can update their website on the fly without having direct access to the source code. Data is stored in tables which can be accessed at any time using queries. Structured Query Language (SQL) is used to manipulate this data. SQL is a standard language that is used in almost all databases [11]. MySQL was chosen for this project as it was the cheaper option for the Liikkunen business, Microsoft SQL was available but it would have been too expensive.

Figure 8 Final Design used the database more than any other page. It displays all the course information, general pictures, dynamic links and banners. Other pages do not use

the database to such a degree. Figure 10 display's the tables that are used for this page and for the course registration.

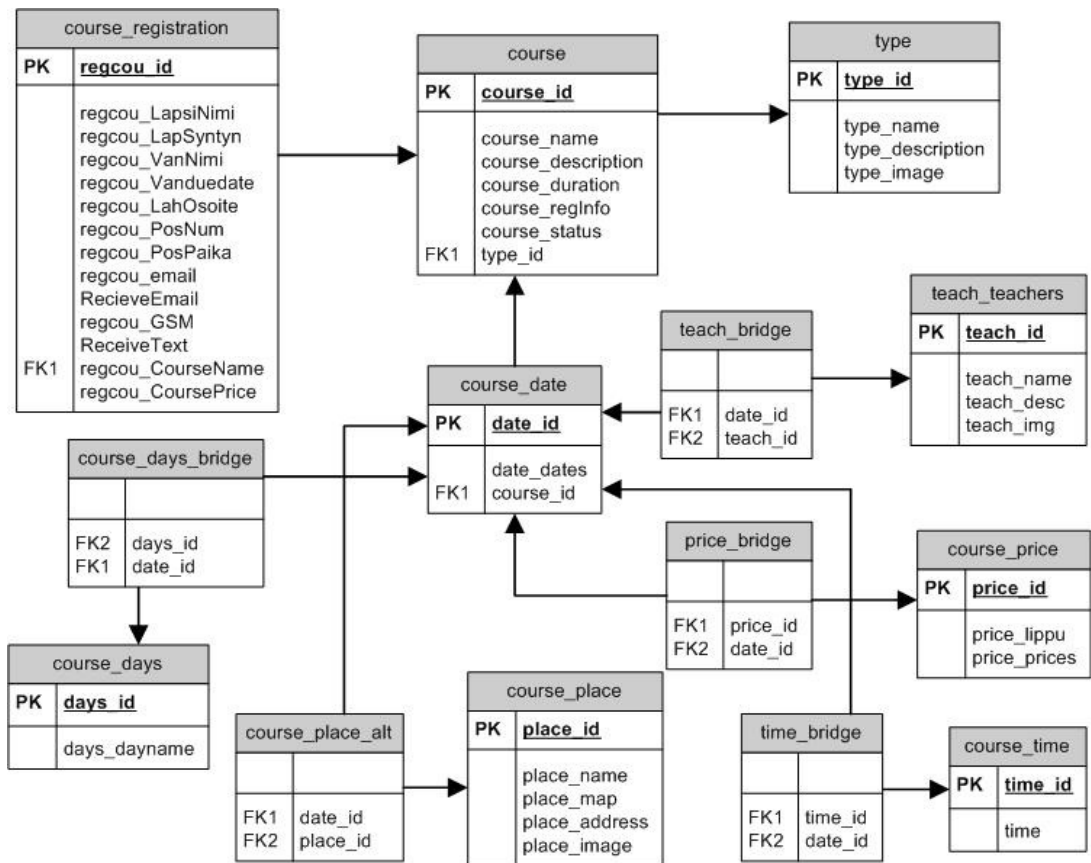


Figure 10. Kurssit.php Database tables

Each of these tables allow for the direct manipulation of the data that is displayed. Each course can have its own teacher, time, date, place and day. If there are multiple dates for a course each can be displayed separately. The course 'Mammajumppa' has such a feature. There are three different dates that the course is running, so instead of having three different courses there is only one. Figure 11 demonstrates this.

Mammajumppa [Ilmoittautuminen](#)

Mammajumppa on kehonhallintajumppa, jossa käytämme apuvälineenä pientä jumppapalloa.

Nostamme sykettä, vahvistamme syviä lihaksia ja tunnin päätteeksi rentoudumme.

Mammajumppa on kurssimuotoinen. Kurssilla on viisi kertaa, joista kaksi voit korvata esteen sattuessa. Voit tietenkin olla mukana tunneilla myös usemman kurssin ajan.

Jumppapallon voit lainata ohjaajalta tai ostaa omaksi ilmoittautumisesi yhteydessä.

Kesto: 55min

Ajankohdat	Paikka	Päivä	Aika	Ohjaajat
04.09.-02.10.2009	MOTIVUS Stockmannilla	Perjantai	18:20	Inka Niitepöld
09.10.-06.11.2009	MOTIVUS Stockmannilla	Perjantai	18:20	Inka Niitepöld
13.11.-11.12.2009	MOTIVUS Stockmannilla	Perjantai	18:20	Inka Niitepöld

Figure 11. Mammajumppa Course

When first coding this page, a separate SQL query was generated for each of the Ajankohdat, Paikka, Päivä, Aika and Ohjaajat. This in turn ended up with more code than what would normally be required, about 6-8 different query statements were generated. The solution found was 'LEFT JOIN'. LEFT JOIN will return all rows from one table even if there are no matches in the second (right) table [12]. Once this query was employed it reduced five different queries into one powerful query. The full query can be seen in appendix 2.

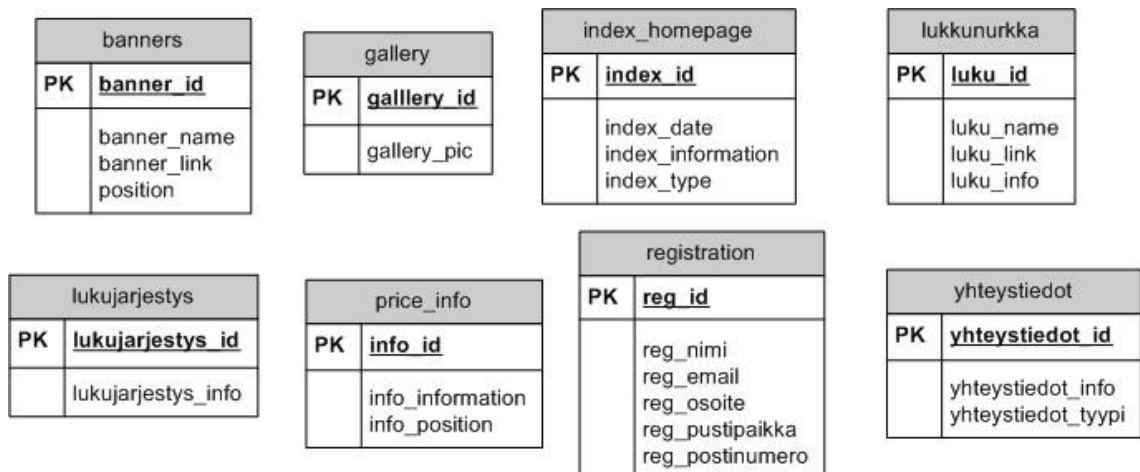


Figure 12. General Database Tables

Figure 12 represents the remaining tables that are used for this site. Some of these tables, such as banners are accessed more often than others. The customer expressed the desire to add more advertisements to the site and wished to have a means to add them. Other tables are page specific; they will only be accessed once their respective page has been called by the user.

3.4 Cookies

Cookies are commonly used in websites. They are used for a variety of purposes such as: user's navigation through your site, authorization, encrypting and deleting [10]. For this site only cookies on one page, Kurssit page, were used. The reason being is that when a user goes to a specific type of course the links on the left of the page are dynamic. They change depending on the type of course selected.

The links needed to be available for the user. When the user views a course the general information disappears and the course specific information replaces it. Also without cookies the links disappear, this would also happen if the user has cookies turned off in their browser.

```

$ID = $_GET['course'];// getting the course ID and keeping it in the browser;
if($ID == 0){

}
else{
    setcookie ('coursetype', $ID, time()+ (3600));
}

```

Code Example 2. Cookies Kurssit.php

For the cookies on this page, code example 2, the course type_id, which is a number, is passed, and it is then stored in a cookie named “coursetype”. This is then held for one hour. The reason for one hour is that there should be no one still on the page after an hour; therefore, setting the expiration longer would not be necessary. Cookies were deemed not to be necessary for any other page since the links do not change and any changes for those pages would be in the database.

3.5 Regular Expressions

PHP supports two kinds of regular expressions, POSIX (Portable Operating System Interface for Unix) and Perl. Regular expressions can be used to match input data according to predefined constraints [9]. The PHP community is now currently moving over to Perl as the standard for Regular Expressions since POSIX has been deprecated as of PHP 5.3 and will be removed from PHP 6.0 [13]. For this project user input had to be checked for problems that could occur; this was then matched to a predefined regular expression.

Various problems were encountered such as

- 1) Individuals had “-“, in either their Forename(s) or Surname(s)
- 2) Individuals using two first names without “-“
- 3) Having multiple names in one field, such as a parent signing in three children into one course or two adults signing into a course with their child, or a combination of both.

The original predefined Regular Expression:

```
(preg_match("/^[a-öA-Ö]+(-[a-öA-Ö]+)?$/", $LapNimi) )
```

This expression could not handle what was necessary for the site. A new expression was created so that it could handle all cases of Finnish names:

```
(preg_match("/^[ a-öA-Ö\+-]+( [a-öA-Ö\-,]+)*$/D", $LapNimi) )
```

3.6 Browser Compatibilities

Browser Compatibility is the rendering of a website on many different browsers (such as Internet Explorer (IE) and Firefox) to see how the site appears. What may work on for one browser may not work for another. When using Cascading Style Sheets (CSS) items such as “a:link” work with all browsers but others such as “max-width” do not. Different browsers have CSS implemented in different ways. IE 7 and 8 both use correctly max-width but IE6 will not, the CSS version for IE6 can be best described as buggy. Not all the usual CSS classes or id’s work. Other examples are float, h1 + p, etc [14].

Throughout coding, site checks had to be made in order to see how the site reacted in different browsers. The browsers used were Firefox 3.5, Internet Explorer 8, Opera and Chrome. The biggest issued was on the Yhteystiedot page. The image that is used did not appear correctly in all browsers. Only in Firefox did it appear correctly. At first div’s and CSS were used to set its position, Firefox was the only success; the other browsers pushed the image below the text. Finally a table was used to set the position.



Figure 13. Internet Explorer - Yhteystiedot.



Figure 14. Firefox – Yhteystiedot.

Further testing includes how browsers render the normal “text” and “colour”. Every browser has its own rendering engine. As can be seen from figure 13 and figure 14 the text and line spacing in IE is larger than in Firefox. Also from the image the image in IE is darker than in Firefox. Since there is no one, common, rendering engine nearly all browsers are using different engines. Firefox uses the Gecko rendering engine, and Trident for IE. Each engine will display the content slightly differently from the other.

3.7 Multilingual Sites

In today's modern Internet not only is English used, but also Finnish, Swedish, Chinese, and other various languages that are Latin and non-Latin based. As the Internet ascends to a multilingual society, websites must also become multilingual, or at least have the capacity to display these languages. There is already the ISO-8859-1 character set which included all Latin based characters but also adds support for Finnish ä and ö, and other characters from other languages such as ü from German. Unfortunately, not all character sets were cross compatible such as ASCII – English characters only or ANSI. Some of these character sets were only available to particular Operating Systems – ANSI for Windows and MacRoman for Apple. Some of the characters in both of these character sets are not available in each other or in ISO-8859-1 [15]. From this mess of different character sets Unicode was born. Unicode provides a unique number to every character – unlike the old old sets which may have different numbers for the same character – so that every Operating System or program will have the exact same character set [16].

While creating the Liikkunen site the character the ISO-8859-1 was used, this encoding has been used on many websites without issues. The predominant issue that arose was during the creation of the administration. This reason will be dealt with in section 4.4; because of this issue the site was changed into Unicode – UTF-8 as the site character set.

When changing over from the ISO-8859-1 to Unicode (UTF-8) the characters were at first not displaying correctly. Instead of ä or ö, ¤ was rendering instead; all other characters – ASCII character set – worked perfectly. It was not until further research that I discovered that Notepad++ – the editor of choice – encoded pages in ANSI. So on top of changing the “charset” in the head of the HTML document, the pages of the site had to be re-encode to UTF-8 so that the Scandinavian letters would render as they should.

Upon re-encoding the pages a new problem was discovered; all the Scandinavian letters were rendering – when the page was viewed in Notepad++ and also when viewed in any browser – were replaced by other characters. Figure 15 represents the new characters after the encoding.



Figure 15. Characters after UTF conversion.

On the left is the new Unicode characters, and the right has the original characters. The only way to have these “new” characters removed was to manually retype all of them for every page. Once this was completed, the pages would render correctly in Firefox or IE.

4 Designing the Administration Interface

4.1 Design

The general design of the administration is virtually the same at the General User Interface. The main differences are: no banners at the right or bottom of the screen, a wider interface and more links at the top of the site. There are several reason for this, it is firstly: since it's the administration, there was no need for the banners due to the fact that only the site owner would see this part of the site. Secondly, a wider interface was needed to allow for the extra information that would be available to the customer so that there would be no table that would be displayed outside the general borders of the administration. Lastly, more links so that the customer can change any part of the site that she wishes. All aspects of the site are at her disposal. Figure 16 displays the general layout for the administration.



Welcome to your frontpage of the Administration Inka

All pages are now fully functional, you can ADD, DELETE, MODIFY everything.
This is the **Button Key** that is used in all the Adding / Modifying forms

[Add new Information](#)

index_date	index_information	index_type		
27/08/2009	Keskiviikkoisin klo 13 alkava vauvajumppa on täynnä. Lisäryhmä 9.9. alkaen klo 13.45. Ilmoittaudu pian!	Päivitykset	Modify	Delete
	<p>Leikkien liikkumaan!</p> <p>Liikkunen tarjoaa odottavien äitien sekä äitien ja vauvojen liikuntaa Motivus Stockmannin kauniissa salissa keskellä kaupunkia sekä muualla Helsingissä.</p> <p>Ajankohtaista:</p> <p>Syksyllä 2009 Motivus Stockmannilla 31.8. alkaen:</p> <ul style="list-style-type: none"> - Mammajumppa - Vauvajumppa "pienet" 2kk-7kk - Vauvajumppa "isot" 8kk-1,5 v. <p>Tervetuloa liikkumaan kanssamme!</p> <p>Ilmoittautumaan pääset kunkin kurssikuvaussivun kautta.</p>	Ajankohtaista	Modify	Delete
13/09/2009	Seuraava Mammajumppa-kurssi (9.10.-6.11.2009) Ilmoittautuminen on alkanut!	Päivitykset	Modify	Delete
20.10.2009	Ilmoittautumislomake on väliaikaisesti pois käytöstä. Ilmoittaudu sähköpostilla toimisto@liikkunen.fi	Päivitykset	Modify	Delete

Figure 16. Liikkunen Administration

As can be seen from figure 16 the general lay has been kept, one of the main reasons for this is that the customer knows that she is on the same site and has not been directed elsewhere on the Internet [1]; also it keeps the continuity of the site flowing. The main reason for having the site wider then on the User Side is to show some of the basic information in the “Kurssit Ilmoittautuminen” page. Figure 17 shows the reasons for having a large width.

The screenshot shows the website 'Liikkunen.fi' with a navigation menu at the top. The main content area is titled 'Course Registration' and 'Vauvajumppa 2-7 kk - Ryhmä 2'. Below the title, it states 'The current number of registered people is: 2'. A table lists the registered users with columns for 'regcou_LapsiNimi', 'regcou_LapSyntyn', 'regcou_VanNimi', 'regcou_email', and 'regcou_CoursePrice'. The table contains two rows of data, each with 'Modify' and 'Delete' links.

regcou_LapsiNimi	regcou_LapSyntyn	regcou_VanNimi	regcou_email	regcou_CoursePrice
Tadhg Clancy	2009 Kes?kuu	Name Surname	name.name@hotmail.com	Kokeilutunti, 12 <input type="checkbox"/> Modify Delete
This-is my-big-name	2009 Hein?kuu	mother-of big-name	bignamesmothers@emailaddress.com	Loppukausi, 9 <input type="checkbox"/> / tunti Modify Delete

Figure 17 Kurssit Ilmoittautuminen.

The biggest contributing factor for having the width is that individuals' emails can show their first.surname@somewhere.com. On top of the email is the number of columns that much be shown. The page will only display certain columns depending on the course. Since Vauvajumppa will have both parent and child, both need to be displayed. The reason for this is that the customer wanted to be able to make quick lists of who is in the course without having to resort to another file for sorting the lists, deleting the excess fields to be left with the same fields presented in figure 17.

One of the main design challenges was deciding on how to proceed with the adding / modifying / deleting of material. One way was to keep everything in the same interface where one proceeds to a new page to edit the site. Keep the current interface and open a new window where changes can take place. I chose the second option, one of the main reasons for doing so is that you can see straight away any changes that take place. If

something is modified and when the page reloads after pressing submit the parent page reloads with the changes while the windows stay open. As stated one can see the changes happen straight away but also one does not have to navigate away from the current page to a new area of the site to do any modification.

4.2 Admin Usage of OOP PHP

As stated in section 3.2 OOP PHP uses classes to call and display the information that is required. There is no difference in between the General User Interface OOP PHP and the administration OOP PHP. A connection is made to the MySQL site through a connection string which holds the username, password, database location and database name. This connection is converted into an object via the mysql class.

For the mysql class first before a query can be sent, can a connection be made to the MySQL server; if successful then next is to determine if the required database can be selected. When both of these are successful a query can be made. As stated earlier, the mysql class created an object from the connection string, the exact same process is applied to the SQL statement / query. Once the desired query is created, the connection string creates an object from the query and sends the query to the MySQL database through the mysql class.

When the query is sent to the database and can be confirmed to be a valid, query the output can be processed in the mysql class. Once the output is determined, it can finally be displayed. Figure 18 demonstrates the process that takes place.

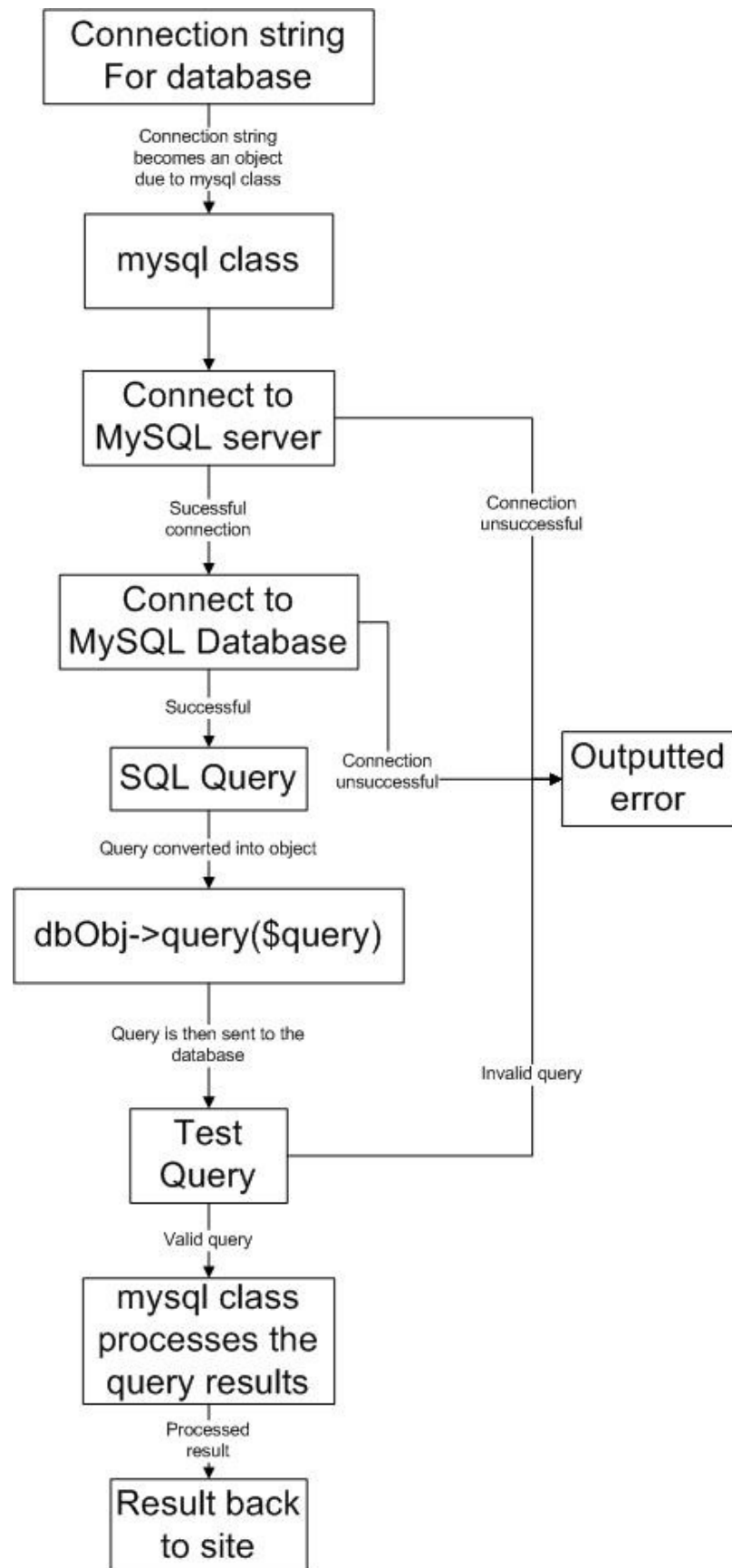


Figure 18. OOP PHP process.

The biggest issue's that were found was in modifying the Input and Form Classes. When creating textareas for the site so that the customer could add or modify information, the textarea was not displaying correctly. Since a textarea needs both opening, and closing tags and not just one self closing tag (as is the input tag), the textarea needs different arguments to the input. Input, when creating it, only needs the length, while a textarea needs clos and rows as input arguments for its creation. Code example 3 and code Example 4 show the differences in how they are created.

```
$pDate = new inputText("date", "30");
$code .= $pDate->getCode();
```

Code Example 3. Creating an Input field

```
$pDate = new InputTextArea("message", "40", "10");
$ code .= $pDate->getCode();
```

Code Example 4. Creating a Textarea

Once the textarea was showing as required, the next issue was how to display information in the textarea that would be modified. The issue was with the Input Class itself and not any where else.

As was stated earlier an Input tag is self-closing, but the textarea tag has both an opening and a closing tag. So, when setting the value for the Input it is naturally in the tag, code example 5 demonstrates this, at first I was trying the same method with the textarea.

```
$response="<input $dis $maxl $reado $nam $typ $val $siz /> \n";
```

Code Example 5. InputText Code from the Input Class

To achieve the favourable outcome for the textarea to display the text that was to be modified, the \$val had to be moved between the textarea tags. Code example 6 displays the final version for the InputTextArea class.

```
$response="<textarea $dis $maxl $reado $nam $typ $rows $cols> $val </textarea> \n";
```

Code Example 6. InputTextArea Code from the Input Class

Along with this change, other values needed to be created, two private values for cols and rows, and the class had to have their own constructor. The full Input Class can be found in appendix 2 figure 2.

For further changes in the site the Form class had to be modified, so that a form name could be passed. Only three lines of code were added, and the constructor had a new value add. This was used so that a piece of JavaScript could be used to enable the customer to use HTML tags. The JavaScript created HTML tags in the textarea so that the customer could define her text. Some of the tags are H1, H2, strong, br and p. Another piece of JavaScript was used for the “€” sign.

4.3 Security

Securing one’s website is just as simple as an on-off switch, such as, secure or not secure. It is a scale on which a site can be measured on. The more secure a site is the more checks need to be done; more code, more MySQL / server checks. More security can also translate into reduced performance for both the user and administrator [16].

Authentication is also a valuable part of website security. The credentials of the user should be checked; this is usually done via a username and password combination. Once this has been inputted, the database can then be checked [17]. Once the user is authenticated, sessions can begin. The session will hold the necessary data that can be used when checking users’ rights on different pages [11].

One of the biggest security risks on the Internet is having one's password stolen. If a hacker gets access to a database, then they have access to all the information present, including users usernames and passwords. One method of preventing this is by Hashing the password. md5() function uses MD5 which is a hash algorithm that is often used for creating signatures. MD5 is considered one-way hashing making it very difficult for the original password to be dehashed [8].

For the General User Interface there is no login data to be retrieved. There are however two different forms that the user can fill out: course registration and normal registration (for news and updates). For the two forms, Regular Expressions are used as the main security precaution. The regular expressions test the user input, so that it represents a certain form. If the data is not of this form, an exception or error is produced and the user must retype their input.

For the administration a login, sessions and authentication are used to check the users details. Also the user's password is hashed for the added security. The user can be limited to only certain features depending on their level on access. The customer has access to the entire site; she can add / modify / delete everything. The teachers on the other hand have only access to the basic course list. They cannot add / modify / delete user data.

4.4 Admin Site Multilingual Problems and Solutions

As stated in section 3.7 the Liikkunen site was converted over from ANSI, file encoding, and the ISO-8859-1, character set, to Unicode. The main reason for this was for comparing Scandinavian characters such as, 'ä' to 'ä', or 'ö' to 'ö'. One of the courses in the Liikkunen site is called "Vauvajumppa 2-7 kk- Ryhmä 2"; unfortunately the Scandinavian "ä" could not be translated. A query was first used to retrieve the name of the course. This result was then used to find the registered people in that particular course.

When tested in the user interface for MySQL, phpMyAdmin, the query works, however when called to the user interface, no results could be found. The database itself had to

be converted to Unico. The collation that was first used was latin1_swedish_ci, but this proved to be inadequate for the needs of the site. There each table and every field had to be converted from latin1_swedish_ci to Unicode_swedish_ci. Only then did the registered user list appear.

When converting the files format from ANSI to Unicode, Notepad++ gives two different options:

- 1) Encode in UTF-8
- 2) Encode in UTF-8 without BOM

If the file was encoded using option 1, Encode in UTF-8, the resultant output file would have an error. Figure 19 shows a file Encoded in UTF-8.

Warning: Cannot modify header information - headers already sent by (output started at E:\xampp\htdocs\Liikkunen\site\Kursnit.php on line 6)

KOTI KURSSIT LUKUJÄRJESTYS
 LUKUNURKKA GALLERIA LIIKUNTAPAIKAT
 HINNASTO REKISTERÖIDY YHTEYSTIEDOT





[Odottava äiti](#)



[Äiti ja vauva](#)



[Lapset ja vanhemmat](#)



[Lapset](#)



[Aikuiset](#)



Odottava äiti

[Chiball](#)
[Kehonhuoltokurssit](#)
[Mammajumppa](#)
[Mammajumppa2](#)

[Yleistä](#)

Odottaville äideille

Liikkuminen on tulevan äidin kannalta ensiarvoisen tärkeää; ei vain fyysisen vaan myös henkisen kunnan vahvistamisessa. Raskauden aikainen liikunta valmistaa kehoasi tulevaan synnytykseen.

Liikunta vahvistaa lihaksia, parantaa ryhtiä ja tasapainoa, auttaa painonhallinnassa, parantaa kestävyyttä, auttaa selkää- ja ilmavai-voihin unohtamatta hyvää mieltä, sitä euforista oloa jumpan jälkeen. Hyvääolon tunteen myötä stressinsietokyky paranee. Liikunnan avulla voit parantaa myös kehontuntemustasi ja valmistautua tuleviin muutoksiin.

Liikunnan merkitys itse synnytyksessä ja palautumisessa on merkittävä, jaksat varmasti paremmin. Liikunnan parista voi myös löytää kaverin, jonka kanssa voi jakaa raskauteen liittyviä kysymyksiä. Kuuntele kehoasi, liiku ja nauti!

Site created by Tadhg Clancy





Figure 19. File Encoded in UTF-8

BOM stands for “Byte Order Mark”; it consists of the character code U+FEFF. It can be used to define the encoding for unmarked files [18]. Figure 8 displays the same file when it is Encoded in UTF-8 without BOM.

An interesting consequence of converting the site and database into Unicode was that the Scandinavian letters in the database appeared differently. But when they are called from the database they appear as normal:

Ä = Ã„, example: Äiti ja vauva = Ã„iti ja vauva
 ä = Ã¤ example: Odottava äiti = Odottava Ã¤iti
 ö = Ã¶ example: myös = myÃ¶s
 õ = Ãµ example: Inka Niitepõld = Inka NiitepÃµld

This, I attributed to Unicode and how it interprets letters from different languages.

4.5 Adding – Modifying – Deleting Data

Knowing how to Add, Modify and Delete data in the administration is a necessity. This section will cover how to Add a full course and then Modify the said course. Figure 20 represents the first look for the Course Information – Kurssit Info.

The screenshot shows the 'Kurssit Info' page on the Liikkunen.fi website. At the top, there is a navigation menu with links: KOTI, KURSSIT ILMOITTAUTUMINEN, KURSSIT TYPPI, KURSSIT INFO, LUKUNURKKA, GALLERIA, LIIKUNTAPAIKAT, LUKUJARJESTYS, BANNERS, HINNASTO, REKISTERÖIDY, OHJAAJAT, YHTEYSTIEDOT, and LOGOUT. The main content area is divided into two columns. The left column, titled 'Kurssit', contains a list of course links: Mammajumppa, Chiball, Kehonhuoltokurssit, Mammajumppa2, Vauvajumppa 2-7 kk - Ryhmä 2, Vauvajumppa 8 kk - 1.5 v, Vauvajumppa 2-7 kk, Perheliikunta 1-2 vuotiaalle, Perheliikunta 3-5 vuotiaalle, Capoeira, and Feldenkrais-tunnit. The right column, titled 'Kurssit Info', contains several sections: 'Add new Information', 'Kurssit: Paikka, Päivä, Aika ja Ohjaajat' with a link to 'Add new Dates', and 'Kurssit: Hinta' with links to 'Add/Modify new Hinta Info' and 'Add new Hinta'.

Figure 20. Kurssit Info page

As one can see from figure 20, the list of courses is present on the left of the screen. These courses are not in alphabetical order but are in course type and course id order.

Clicking on any of the links will display all the selected course information that is present in the database.

When adding a new course, the user is brought through four different stages. Each stage will ask you for different information pertaining to the course:

Stage 1: Course Name, Description, Duration, Type and Status.

Stage 2: Date, Place, Day, Time, Teacher

Stage 3: Payment Information

Stage 4: Price(s)

The reason for having so many different stages is that the owner may not know certain aspects of the course until later on. Example: When will a course start? Until it is known only Stage 1 would need to be completed. Once the Date, Place, etc are known, then the course can be filled with all the necessary information. The only stage where one may have to revisit is Stage 4. Some courses may only have one price while others may have many different pricing schemes, therefore, it was decided that only one price could be inserted at a time.

For the Modifying of a course the same process would be undertaken as in Adding. Different stages would represent the various aspects of the course. There would only be one difference between them, Stage 4 would not exist. The reason being is that it would be normal to add / delete a new price instead of modifying one. Figure 21 shows when a course is fully developed and displayed.



Kurssit

- [Mammajumppa](#)
- [Chiball](#)
- [Kehonhuoltokurssit](#)
- [Mammajumppa2](#)
- [Vauvajumppa 2-7 kk - Ryhmä 2](#)
- [Vauvajumppa 8 kk - 1.5 v](#)
- [Vauvajumppa 2-7 kk Perheiliikunta 1-2 vuotiaalle](#)
- [Perheiliikunta 3-5 vuotiaalle](#)
- [Capoeira](#)
- [Feldenkrais-tunnit](#)

Kurssit Info

[Add new Information](#)

Mammajumppa

course_description	course_duration	course_regInfo	course_status		
<p>Mammajumppa on kehonhallintajumppa, jossa käytämme apuvälineenä pientä jumppapalloa.</p> <p>Nostamme sykettä, vahvistamme syviä lihaksia ja tunnin päätteeksi rentoudumme.</p> <p>Mammajumppa on kurssimuotoinen. Kurssilla on viisi kertaa, joista kaksi voit korvata esteen sattuessa. Voit tietenkin olla mukana tunneilla myös usemman kurssin ajan.</p> <p>Jumppapallon voit lainata ohjaajalta tai ostaa omaksi ilmoittautumisesi yhteydessä.</p>	55	<p>Maksa kurssimaksu Liikkusen pankkitilille. Tilinumero on Sampo-pankki 800011-71333405 / Liikkunen (löydät tilinumeron jatkossa kohdasta Yhteystiedot). Ota kuitti maksusta mukaan tunnille. Liikuntaseteleillä maksat suoraan ohjaajalle.</p> <p>Kun tulet tunnille, Motivuksen tiskillä ei tarvitse ilmoittautua. Lippuasiat hoidetaan Liikkusen ohjaajan kanssa tunnin alussa.</p> <p>Esteen sattuessa voit korvata kaksi kurssikertaa toisen vastaavan kurssin tunneilla tai vauvajumppassa. Korvauseroille ilmoittaudut sähköpostilla: asiakaspalvelu@liikkunen.fi.</p>	full		Modify Delete

Kurssit: Paikka, Päivä, Aika ja Ohjaajat

[Add new Dates](#)

Ajankohdat	Paikka	Päivä	Aika	Ohjaajat	
04.09.-02.10.2009	MOTIVUS Stockmannilla	Perjantai	18:20	Inka Niitepöld	Modify Delete
09.10.-06.11.2009	MOTIVUS Stockmannilla	Perjantai	18:20	Inka Niitepöld	Modify Delete
13.11.-11.12.2009	MOTIVUS Stockmannilla	Perjantai	18:20	Inka Niitepöld	Modify Delete

Kurssit: Hinta

[Add/Modify new Hinta Info](#)
[Add new Hinta](#)

price_lippu	price_prices	
5 krt kortti	65€ (pallo omaksi)	Delete
5 krt kortti	55€ (lainapallo)	Delete
10 krt/kortti	100€	Delete

Figure 21. Full course information

The same processes (add / modify) can be used for all the information present in the administration. The only different section to the rest is the Kurssit Ilmoittautuminen. Here one can see the basic information that is there for each course registrant, so the customer can see who has registered and can make a quick reference list from those who are registered. There is a link above and below the table that houses the information. This link opens in a new window and will display all the information that the people have registered: Name, email, address, phone, etc.

5 Conclusions

The aim of the project was to redesign the Liikkunen website so that the owner could update the site without having to contact a third party. The objectives were met. The old site needed to be updated to allow for standardised linking system and logo placement. The lack of these two basic needs prevented people from knowing where they were on the site and reduced the usability. Also the extra features which could confuse the user are no longer present. The new site design presented the user with a standard interface, for all pages, and it conforms to website standards.

The administration allows the customer to update the site in all aspects. It also increases the work efficiency of the owner as they do not have to deal with emails detailing user data for the course, because all data is presented in structured tables. The administration also gives the teachers the chance to retrieve the course lists without contacting the owner.

The site is robust enough to handle the addition and modification of any new course that the owner can consider. It also enables courses to have many different dates displayed; unfortunately this is also a weakness. Due to the manner of construction, a course cannot be split into its different dates. A new course would have to be constructed for this new date. Another weakness of the site is that the owner cannot create a new page; an outside party would have to be employed to do so. To further increase productivity in the creation process a prebuilt content management system (CMS) could have been used. This would have reduced the time taken for the site to be built.

References

- 1 Krug S. Don't make me think: a common sense approach to web usability. Berkeley, CA: New Riders Publishing; October 2000.
- 2 Microsoft. What is VBScript? [online]. United States Microsoft.
URL: <http://msdn.microsoft.com/en-us/library/1kw29xwf%28VS.85%29.aspx>.
Accessed 16 October 2009.
- 3 Garrison G. How to make a website's logo pop [online]. Los Angeles. Five Finger Coding.
URL: <http://www.fivefingercoding.com/web-design/how-to-make-a-website-logo-pop>. Accessed 5 November 2009.
- 4 Bluejay M. Web design tips [online]. Austin, TX. Website Helpers; 2001 updated
February 2009.
URL: <http://websitehelpers.com/design/>. Accessed 3 November 2009
- 5 Prismo. Evaluation display and window survey [online]. Switzerland: Prismo
URL: <http://www.prismo.ch/surveys/evaluation.php>. Accessed 19 August 2009.
- 6 Steam. Steam hardware survey: September 2009 [online]. Washington: Steam; September 2009
URL: <http://store.steampowered.com/hwsurvey/>. Accessed 19 October 2009.
- 7 Prismo. Notebook LCD display comparison [online]. Switzerland: Prismo.
URL: <http://www.prismo.ch/comparisons/notebook.php>. Accessed 19 August 2009.
- 8 The PHP Group. History of PHP [online]. The PHP Group.
URL: <http://fi2.php.net/manual/en/history.php.php>. Accessed 12 October 2009
- 9 Gilmore W. Beginning PHP and MySQL: from novice to professional, 3rd ed. Berkeley, CA: Apress; 2008.
- 10 Converse T. Park J. Morgan C. PHP5 and My SQL Bible. Indianapolis, IN: Wiley Publishing, Inc; 2004.
- 11 Davis M. Phillips J. Learning PHP and MySQL, 2nd ed. Sebastopol, CA : O'Reilly; 2007
- 12 W3Schools. SQL LEFT JOIN Keyword [online]. W3Schools
URL: http://www.w3schools.com/sql/sql_join_left.asp. Accessed 01 August 2009.


- 13 The PHP Group. `ereg` [online]. The PHP Group.
URL: <http://fi2.php.net/manual/en/function.ereg.php>. Accessed 13 August 2009
- 14 Westciv Wiki. CSS compatibility for current browsers [online]. Westciv Wiki
URL:
http://westciv.com/wiki/CSS_compatibility_table_for_current_browsers#Class.
Accessed 23 October 2009.
- 15 Wood A. Differences between ANSI, ISO-8829-1 and MacRoman character set [online].
URL: <http://www.alanwood.net/demos/ansi.html>. Accessed 24 October 2009
- 16 Ullman L. PHP 6 and MySQL 5. Berkeley, CA: Peachpit Press: 2008
- 17 Schlossnagle G. Advanced PHP Programming. Toronto, Ontario: Sams Publishing: 2004
- 18 Unicode. Inc. UTF-8, UTF-16, UTF-32 & BOM [online]. Unicode Inc.
URL: http://unicode.org/faq/utf_bom.html#bom1. Accessed 11 October 2009.

Appendix 1:

Further Examples of Linking Discrepancies

FITMAMA

Katso esittelyvideo:



MOTIVUS

Liikkuminen on tulevan äidin kannalta ensiarvoisen tärkeää; ei vain fyysisen vaan myös henkisen kunnon vahvistamisessa. Raskauden aikainen liikunta valmistaa kehosi tulevaan synnytykseen. Liikunta vahvistaa lihaksia, parantaa ryhtiä, auttaa painon hallinnassa, parantaa kestävyyttä, auttaa selkäreisä ja ilmavaihtoihin unohtamatta hyvää mieltä, sitä euforista oloa jumpan jälkeen. Näin myös stressinsietokyky paranee ja uskallanpa väittää, että on helpompi ymmärtää raskauden aikana tapahtuvia muutoksia kehossasi. Erityisen tärkeänä pidän lantionpohjan lihasten vahvistamista nopeus- ja voimaharjoittelulla, kestävyysharjoituksia unohtamatta. Lantionpohjan lihakset ovat äärimmäisen tärkeät.

Liikunnan parista voi myös löytää kaverin, jonka kanssa voi jakaa raskauteen liittyviä kysymyksiä.

Liikunnan merkitys itse synnytyksessä ja palautumisessa on merkittävä, jaksat varmasti paremmin. Kuuntele kehoasi, liiku ja nauti!

LIKKUSESSA iloitaan liikkumisesta, tervetuloa nauttimaan FITMAMA tunneista.

TERVETULOA MUKAAN HYVÄNMIELIEN TUNNEILLE!

FitMama-aikataulut tässä
Ilmoittaudu tunnille tästä

Käytännön asioita Motivuksessa paikan päällä:
Motivuksen tiskillä ei tarvitse ilmoittautua. Tullaan suoraan

Figure 1. jumpat_mamma.asp (full)



Figure 2. lukujärjestys links difference.



Figure 3. lukujärjestys – lukujärjestykset.asp (full)

Appendix 2:

Various Code.

```

$query="SELECT
        course_date.date_dates,
        course_place.place_name,
        course_days.days_dayname,
        time.time,
        course_teachers.teach_name
FROM
        course_date
LEFT JOIN
        course_place_alt
ON
        course_date.date_id = course_place_alt.date_id
LEFT JOIN
        course_place
ON
        course_place_alt.place_id = course_place.place_id
LEFT JOIN
        course_days_bridge
ON
        course_date.date_id = course_days_bridge.date_id
LEFT JOIN
        course_days
ON
        course_days_bridge.days_id = course_days.days_id
LEFT JOIN
        time_bridge
ON
        course_date.date_id = time_bridge.date_id
LEFT JOIN
        time
ON
        time_bridge.time_id = time.time_id
LEFT JOIN
        teach_bridge
ON
        course_date.date_id = teach_bridge.date_id
LEFT JOIN
        course_teachers
ON
        teach_bridge.teach_id = course_teachers.teach_id
WHERE
        course_id = $ID ;
";

```

Code Example 1. Source Code for using LEFT JOIN to reduce five queries to one single query – Kurssit.php

```

<?php
abstract class Input{
    protected $accept;
    protected $align;
    protected $alt;
    protected $checked;
    protected $disabled = false;
    protected $maxlength;
    protected $readonly = false;
    protected $size;
    protected $src;
    protected $type;
    protected $value;
    protected $cols;
    protected $rows;

    function __construct($nType, $nName) {
        $this->type = $nType;
        $this->name = $nName;
    }

    public function setDisabled(){
        $this->disabled=true;
    }

    public function setMaxlength($len){
        $this->maxlength=$len;
    }

    public function setReadonly(){
        $this->readonly=true;
    }

    public function setValue($val){
        $this->value=$val;
    }
}

class InputText extends Input {

    function __construct($nName, $nSize) {
        parent::__construct("text", $nName); // call the constructor of the Input class
        $this->size = $nSize;
    }

    public function getCode(){
        if ($this->disabled)
            $dis = 'disabled="disabled"';

        if (isset($this->maxlength))
            $maxl = 'maxlength="'. $this->maxlength. '"';

        if ($this->readonly)
            $reado = 'readonly="readonly"';

        if (isset($this->value))
            $val = 'value="'. $this->value. '"';

        $nam = 'name="'. $this->name. '"';
        $typ = 'type="'. $this->type. '"';
    }
}

```

```

    $siz = 'size="'. $this->size. "'";

    $response=<"input $dis $maxl $reado $nam $typ $val $siz /> \n";
    return $response;
  }
}

class InputTextArea extends Input{

    function __construct($nName, $nCols, $nRows){
        parent::__construct("textarea", $nName);
        $this->cols = $nCols;
        $this->rows = $nRows;
    }

    public function getCode(){

        if ($this->disabled)
            $dis = 'disabled="disabled"';

        if (isset($this->maxlength))
            $maxl = 'maxlength="'. $this->maxlength. "'";

        if ($this->readonly)
            $reado = 'readonly="readonly"';

            $nam = 'name="'. $this->name. "'";
            $typ = 'type="'. $this->type. "'";
            $rows = 'rows="'. $this->rows. "'";
            $cols = 'cols="'. $this->cols. "'";
            $val = $this->value;

    $response=<"textarea $dis $maxl $reado $nam $typ $rows $cols> $val </textarea> \n";

    return $response;
    }
}

class Submit extends Input {

    function __construct($nName, $nValue) {
        parent::__construct("submit", $nName); // call the constructor of the Input class
        $this->value = $nValue;
    }

    public function getCode(){
        $nam = 'name="'. $this->name. "'";
        $typ = 'type="'. $this->type. "'";
        $val = 'value="'. $this->value. "'";

        $response=<"input $nam $typ $val /> \n";
        return $response;
    }
}

class Hidden extends Input {

    function __construct($nName, $nValue) {
        parent::__construct("hidden", $nName); // call the constructor of the Input class
        $this->value = $nValue;
    }

    public function getCode(){
        $nam = 'name="'. $this->name. "'";

```

```
$typ = 'type="'. $this->type.'";
$val = 'value="'. $this->value.'";

$response="<input $nam $typ $val /> \n";
return $response;
}
}

class Passwd extends Input {

function __construct($nName, $nSize) {
    parent::__construct("password", $nName); // call the constructor of the Input class
    $this->size = $nSize;
}

public function getCode(){
    $nam = 'name="'. $this->name.'";
    $typ = 'type="'. $this->type.'";

    $response="<input $nam $typ /> \n";
    return $response;
}
}

?>
```

Code Example 2. Full Version of Input class.