OULU UNIVERSITY OF
APPLIED SCIENCES

Yini Wang

DEVELOPING A MOBILE SITE USING KOODIVIIDAKKO CONTENT MANAGEMENT SYSTEM TOOL – SIVUVIIDAKKO

DEVELOPING A MOBILE SITE USING KOODIVIIDAKKO CONTENT MANAGEMENT SYSTEM TOOL – SIVUVIIDAKKO

Yini Wang
Bachelor’s thesis
Autumn 2013
Business Information Technology
Oulu University of Applied Sciences

ABSTRACT

Oulu University of Applied Sciences

Degree in Business Information Technology

Author(s): Yini Wang
Title of Bachelor's thesis: Developing a mobile site using Koodiviidakko content management system tool − Sivuviidakko
Supervisor(s): Tapani Alakiuttu
Term and year of completion: Autumn 2013 Number of pages: *31 + 2 appendices*

Antell is a nationally renowned family company which has business of staff restaurant, café and bakery. In 2012 they decided to renew their website. An advertisement agency has ~~the~~ designed the new web site look. The desktop web site is implemented with a Content Management System called Sivuviidakko, which is a product of Koodiviidakko used to manage the content and appearance of a website.

Koodiviidakko is a young and agile company specialized in digital marketing and communication software. The thesis author who works at Koodiviidakko is responsible for implementing the mobile site of Antell. The design of mobile site comes from the same advertisement agency which designed the new desktop website.

The whole process of making Antell's mobile site is illustrated in this bachelor's thesis. First the whole mobile site was analyzed in order to plan arrangement of the content and implementation. Next the implementation will be descripted in details from aspects of CSS, HTML, JavaScript and jQuery. Last the testing is explained as well and some examples of bugs will be illustrated. However, this thesis will not focus on very technical details, but the main process of how to make a mobile site in general.

The main result of this bachelor's thesis is that Antell has a new mobile site.

Keywords: web development, mobile site, Content Management System, HTML, CSS, JavaScript , jQuery, Google Maps.

CONTENTS

1 INTRODUCTION

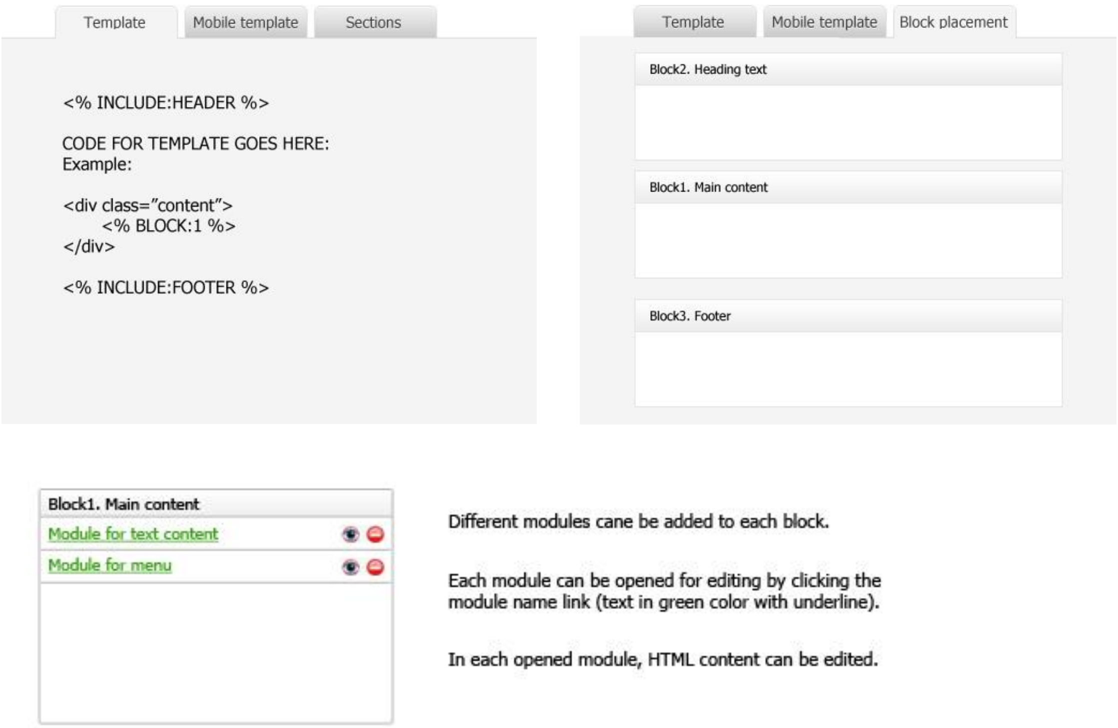
The background of thesis is descripted in this chapter. There is also brief introduction of related case companies and software which is used for making the mobile website.

1.1 Koodiviidakko and Sivuviidakko

"Liana Technologies is a young and agile company specialized in digital marketing and communication software. Our clients are professionals in digital marketing, communications and advertisement - our mission is to give them the best possible tools in the form of digital Liana product family." – The description from company's web site (Liana technologies 2012, date of retrievals 18.8.2013). Liana Technologies is the registered English company from original Finnish company Koodiviidakko.

Product named Sivuviidakko from Koodiviidakko's product family is a versatile and analytics friendly Content Management System ('CMS' is used as abbreviation of Content Management System in the following text) which is used to manage the content and appearance of a website. The CMS separates the content and layout using templates, and content can be added by using numerous different modules with multiple features, and for each module there is "Settings" section where the settings can be defined for the current module, and a "Copy settings" section is for copying the settings, appearance or content from some other modules with same type. Each module uses a default layout for the output, but there is another section called "Appearance" where technical people can add some customized layout for module, in this way the layout is really easily customized.

A template is defined under "Template". A "Block placement" for each template shows how this template will display the content; this can be seen from Figure 1. The template is made up with one or several blocks, and the page which uses this template, under its "Sections" tab, the appearance will be the same as that of "Block placement". Then site manager can add modules into different blocks. There is a mobile template for each template, which will be used only when a mobile device is detected.



```
<% INCLUDE:HEADER %>

CODE FOR TEMPLATE GOES HERE:
Example:

<div class="content">
      <% BLOCK:1 %>
</div>

<% INCLUDE:FOOTER %>
```

Different modules cane be added to each block.

Each module can be opened for editing by clicking the module name link (text in green color with underline).

In each opened module, HTML content can be edited.

FIGURE 1. A sample CMS template and block

In the area of the template, the basic structure of the page is defined. Usually include will be used within template, one include can be used among different templates for avoiding the repeated codes. In a general saying, templates and includes are the place for technical people, whereas the client or site manager should only change the content under individual page's "Sections" tab. The content will be added to each page by different modules.

1.2 Project commission

I work as a front-end web developer at Koodiviidakko. At the time I was going to start my Bachelor's thesis and I wished to write about mobile website since there is more and more users viewing website on their mobile devices. It is not good user experience to view a website with everything zoomed out into tiny size which make the whole site very difficult to read, thus a mobile site is needed. Then there is a case of mobile site for Antell commissioned my company, I thought it would be a very proper and interesting project for my thesis, so I took this case and worked with the mobile site.

The desktop version website has already been finished, and the design of the mobile site came from an advertisement agency, so what I need to do is basically the implementation of the mobile website.

The mobile site will not have as much content as the desktop version site and the design will be simplified for a cleaner view, however there will be some new features added aiming at mobile devices, for example the detection of user geo location and automatically searching for the closest cafés and restaurants.

When the mobile site was almost ready to be published, I started writing my thesis as I thought that was a good time when I still had clear memory of every detail about the whole process of making the mobile site.

2 MOBILE SITE IN GENERAL

” Over the past few years, mobile web usage has considerably increased to the point that web developers and designers can no longer afford to ignore it.” (Raasch.J 2010, How to build a mobile website, date of retrievals 18.3.2013) However, more difficulties are introduced into mobile development because of the special properties of mobile devices. For example, different smartphones have different operating systems; it can be seen from the chart below, the most popular mobile operating systems nowadays are Android and iOS; BlackBerry OS, Windows Mobile and Symbian also has its own certain share of the market which cannot be ignored.

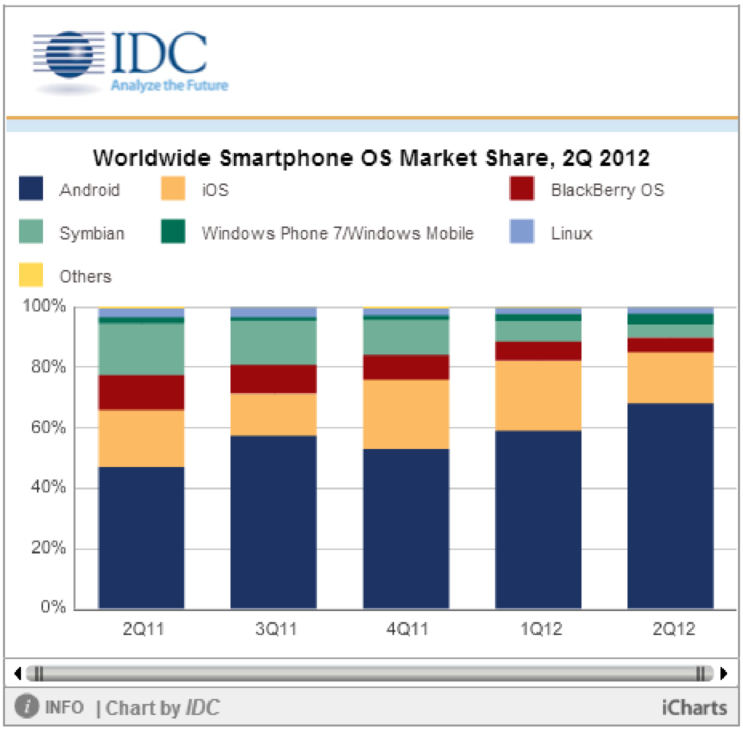


FIGURE 2. Worldwide Smartphone OS Market Share, 2012
(Raasch.J 2010, How to build a mobile website, date of retrievals 18.3.2013)

Thus it is not enough to be only cross-browser in mobile development; it has to be cross-platform as well. In addition to that, any number of browsers can be installed to mobile devices as well, for example there is the native Android browser installed by default on Android devices, but user could also install Opera Mini & Opera Mobile or Firefox Mobile or even Chrome for Mobile (available on Android 4.0, Ice Cream Sandwich or later). That makes a mobile website even more difficult to be compatible in different browser on varied mobile devices.

Moreover, mobile devices have a significantly reduced screen size which limits the content display on mobile devices and the same will apply to the website layout design. In general, a mobile website should have a very clean and tidy layout with less content than corresponding desktop version website.

2.1 Comparison between mobile and desktop version website

In fact, usually the mobile site will be built on basis of desktop site. Therefore there is no need to rewrite everything from the start, but just make the changes for mobile which is different on desktop site and the comparison will do great help to figure out the parts where modification is needed.

2.1.1 Layout and content

Since most mobile devices have much smaller display than pc or desktop, so a solution needs to be figured out for making the content fits the small screen on mobile devices, which could be to replace the multi-columns template by single-column template. In most cases mobile users choose to view website by portrait view - the vertically screen display which make it even more narrow width for displaying the content, thus it will be better to display the content in one column so that the content will not be squeezed too much making it is easier for users to read.

The primary goal of mobile style sheets is to alter the layout for a smaller display. In order to achieve this goal, some unnecessary content should be hidden from mobile website for saving space. It is important to figure out what key pieces of information your visitors will probably be looking for, because mobile users may not be happy to view tons of content through their mobile phone, they have to scroll an extremely long page down to bottom to find the information that they needed. What they may want is just a simple and clean website where they can find the important information easily and quickly. That is one reason why usually there is less content on mobile website than desktop version website.

Another goal of mobile style sheets is to reduce the bandwidth. Usually the mobile networks are slower so if there are too many large images on website, it will take quite much time for loading those images and the user has to wait for long time to open the website. Sometimes if the network situation is poor then the load of image may fail. Neither of that can be considered as good user experience, so it is wise to remove any large background images.

If there is a need to play some videos on your mobile site, it is better not to use Adobe Flash Player at all since it is not supported by any iDevices (iPhones, iPads or iPods), or any Android devices which has version beyond Android 4.0.x. Whereas HTML5 video tag is a recommended alternative to Flash.

2.1.2 Design

When it comes to design part, in most cases a mobile site has a simpler design than the standard site. That is also due to the smaller screen size for mobile devices. Try to minimize the number of images for mobile site design but remain the key content from standard site so that the consistency will be kept. Although the screen size for most mobile devices is comparatively small, certain padding should still be added around content for a better readability.

Many mobile devices have touchscreen interfaces, so try to design with that in mind. Trying to click on tiny, barely-visible links is very annoying, and having to zoom in every time you want to click on something doesn’t make much better. To solve that problem, making the clickable area around your links a little larger, for example with more padding added around it. For some links which have certain purpose it can be considered also to make a button instead.

For navigation, many websites use a completely different design for mobile site. For example there are navigation links placed horizontally on standard site, if this design is just copied to mobile site, when the number of navigation links is not that big then it may fit the small size screen, but if there are too many links, then the navigation will look quite messy and links will be squeezed ugly. However if each of those navigation links are placed in a separate row, and some style is added for links, then the navigation will look well-organized. If the navigation area occupies too much space, consider hiding it at first, and only opening it when user clicks certain “switch” button for the navigation. One example of Rimmel London website is shown in the Figure2 underneath: the navigation will be hidden when page is loaded, but user can open/close by clicking the dropdown button with text “Menu”.

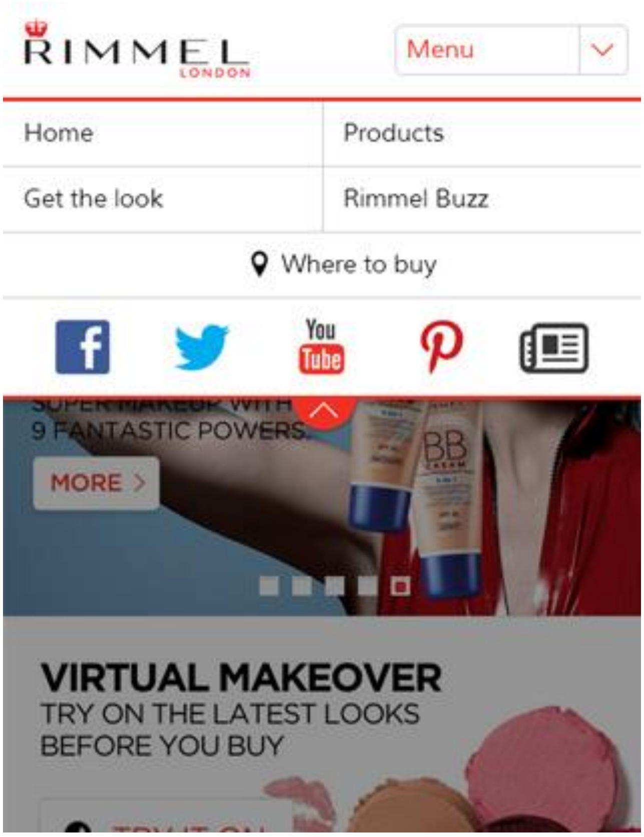


FIGURE 3. Rimmerl London mobile site screenshot (RIMMEL UK mobile site, date of retrievals 19.3.2013)

On mobile site, option should be offered for user to choose whether stay with mobile view or go back to standard website. That could just simply be a button or a link.

If a page includes lot of content that requires scrolling, then a “Back to top” button will be needed, or the header area of the page should stay always at the top in a fixed position.

2.2 Mobile frameworks

“Framework is a library of pre-written JavaScript controls, functions and methods that make it easier for the developer to quickly and accurately produce cross browser compliant code.” This is the definition from Leo Lanese posted on his developer blog which illustrates quite much about what a framework can do for you (Lanese.L 2012, What can a Mobile Framework do for you, date of retrievals 19.4.2013).

One thing needs to be mentioned is that the mobile frameworks discussed in this thesis use~~s~~ HTML, CSS and JavaScript as main development languages, can ~~also~~ be called as JavaScript mobile frameworks. Those JavaScript mobile frameworks are usually used for mobile website and light mobile projects. There are some mobile frameworks that use C++ as main development language, for instance Apache Flex, which is mostly used for developing mobile applications.

Why use mobile framework? Some common features of mobile frameworks may answer that:

Today many of the mobile devices have a touchscreen, which makes the User Interface on touchscreen very different with that on untouchable screens. For example the button should be big enough so that it can be easily clicked, then when user clicked the button, the appearance of the button needs to be different in order to make sure that users can see that they do clicked the button. All the UI elements should make sense and be handy for users to use, that is one of those why it is good to use a framework is that mobile frameworks provide standard UI elements and event-handling specifically for mobile device platforms.

Just like a standard website should be browser compatible, a cross-platform mobile site will help win more users from different platforms. Most of those mobile frameworks have pre-written codes which are cross-platform.

Because of current bandwidth limitations, a stronger emphasis on lowering file size is placed into mobile web development frameworks.

Most mainstream mobile devices have web browsers that support HTML5 and CSS3, so mobile web development frameworks take advantage of new features available in these upcoming W3C specifications for a better user experience.

When to make a decision of choosing one from numerous mobile frameworks, a comparison through different aspects among different frameworks may be needed. One good website is Mobile Frameworks Comparison Chart (Markus.F. Mobile Frameworks Comparison Chart, date of retrievals 19.4.2013), where many mobile frameworks are listed and compared from three major aspects: platform, target and development language. There is even a wizard for searching the suitable framework according to your requirements; the filter is detailed in different categories, for instance it asks the support for hardware features like whether it should support for camera or Bluetooth or geolocation etc.

If searching for top ten mobile frameworks on Google then there will be lots of results. From most of those results ten mobile frameworks will be listed with a brief description of each framework. Those so called top ten frameworks are varied in different results probably because the different standard is used for the rank. Summarizing roughly from those results, the most popular mobile frameworks are: jQuery Mobile, Appcelerator Titanium, The M-Project, Jo, xui.js, EmbedJS, zepto.js, DHTMLX Touch, Mobilize.js and ChocolateChip Mobile (Gube.J, 2011, Top 10 Mobile Web Development JavaScript Frameworks, date of retrievals 20.4.2013).

2.2.1 jQuery Mobile

According to the introduction from jQuery Mobile website, “jQuery Mobile is a user interface framework based on jQuery that works across all popular phones, tablet, e-reader, and desktop platforms. Built with accessibility and universal access in mind, we follow progressive enhancement and responsive web design (RWD) principles. HTML5 Markup-driven configuration makes it easy to learn, but a powerful API makes it easy to deeply customize the library.” (jQuery Foundation. jQuery Mobile 1.3.2, date of retrievals 9.9.2013). Currently the latest version of jQuery Mobile is version 1.3.2

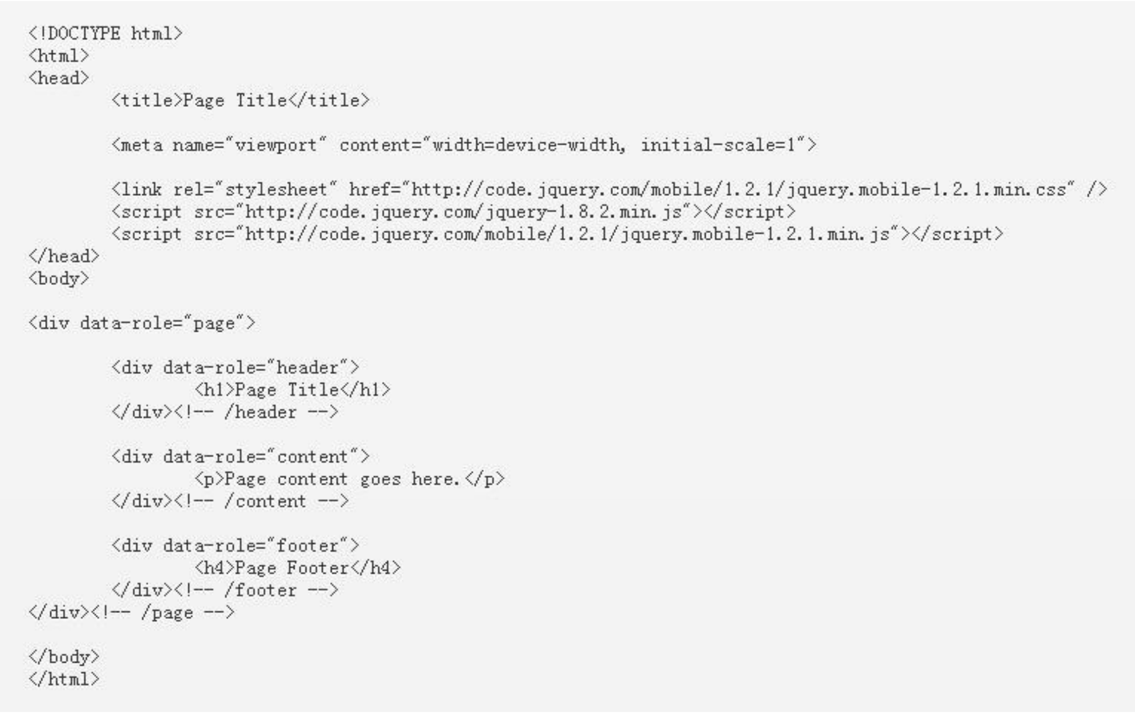
jQuery is a very well-known JavaScript library which has been used in so many standard websites, jQuery Mobile framework is developed based on jQuery, so the framework will benefit~~s~~ from the steady and easy-to-use jQuery core, its touch-friendly UI widgets and thousands of plug-ins based on jQuery.

Moreover, one of the highlight features of jQuery Mobile is Powerful Ajax-powered navigation system. It enables animated page transitions while maintaining back button, bookmarking and clean URLs through pushState plugin.

In addition to that, it is easy to get started with using jQuery Mobile, as shown in Figure 3 jQuery Mobile Standard Boilerplate Template; it basically is just a standard HTML template with few certain requirements.

At first, page should start with a HTML5 “doctype”, no worries for those old devices with browsers which do not support HTML5, because the HTML5 “doctype” and all other HTML5 attributes will be safely ignored.

Secondly a “viewport” meta tag is needed. That is because without a “viewport” meta tag, many mobile browsers will set a virtual page width around nine hundred pixels by default for making it look well with original standard website, which will make the page look too zoomed-out with small mobile screen. Nevertheless if a “viewport” meta tag is set with the content shown in Figure 3 (“width=device-width”), then the page width will be set in pixel as same as that of the device screen. Since there is code like “initial-scale=1” in the content of “viewport” meta tag, then the page is set without any zoom by default.



```
<!DOCTYPE html>
<html>
<head>
        <title>Page Title</title>

        <meta name=“viewport” content=“width=device-width, initial-scale=1”>

        <link rel=“stylesheet” href=“http://code.jquery.com/mobile/1.2.1/jquery.mobile-1.2.1.min.css” />
        <script src=“http://code.jquery.com/jquery-1.8.2.min.js”></script>
        <script src=“http://code.jquery.com/mobile/1.2.1/jquery.mobile-1.2.1.min.js”></script>
</head>
<body>

<div data-role=“page”>

        <div data-role=“header”>
                <h1>Page Title</h1>
        </div><!-- /header -->

        <div data-role=“content”>
                <p>Page content goes here.</p>
        </div><!-- /content -->

        <div data-role=“footer”>
                <h4>Page Footer</h4>
        </div><!-- /footer -->
</div><!-- /page -->

</body>
</html>
```

FIGURE 4. jQuery Mobile Standard Boilerplate Template

Then the last thing to fill out inside head tag is the references to jQuery, jQuery Mobile and the mobile theme CSS. It is the easiest and maybe a very good way to link the files which are hosted on jQuery for the best performance.

After that, let us move to the page content. A single page is identified by a tag inside body tag with data-role="page" and usually a div tag is used. That is like a container for the whole page. Inside that page container, any valid HTML markup can be used. There are many options for data-role of the element, which can be chosen depending on the purpose of the element.

Although mobile framework is very helpful and functional, it is not a rule that you have to use it in all cases. Those frameworks are usually used in web-based applications, for sure it can be used in website as well, however in that case, situation may differ on several factors, for example the design of the website: if the design is very demanding and customized, and not so many complicated functions are needed, then it may be wiser to just not use framework so that lots of time can be saved from changing default style sheet and also the structure of HTML code could be more flexible as well. Actually I do not use jQuery Mobile or any other mobile framework for Antell mobile site, it may help having a better understanding if you read from next chapter of Antell site analysis.

2.2.2 Sencha Touch

Similar with jQuery Mobile, Sencha Touch is a framework for mobile web application based on latest HTML5 and CSS3. Sencha Touch 2.2 is the latest version for now.

"With over 50 built-in components, state management, and a built-in MVC system, Sencha Touch 2 provides everything you need to create immersive mobile apps. (Sencha Inc. Sencha Touch 2013 Who's using)". From description on Sencha Touch official page, the highlighted advantage of this framework is making the web application look like it is native. That can be implemented by using Sencha SDK Tools. "Sencha SDK Tools give you the best of both worlds, providing a way to seamlessly "wrap" your web app in a native shell (Sencha Inc. Sencha Touch 2013 Features)".

Besides that, an updated feature introduced in Sencha Touch 2 of smoother scrolling and animations is very useful and helpful for mobile web applications. Fluid animations and scrolling effects make the interface more user-friendly and create good user experience at the same time.

On demo page of Sencha Touch official website, there is an application named The Kitchen Sink which showcases all of the UI widgets, data features, animations, themes, and touch events offered by Sencha Touch. Each live example page even has a “Source” button where the source code of example can be easily displayed. The Kitchen Sink offers a simple and quick way of getting started of learning Sencha Touch; moreover it can be previewed not only on phones and tablets but also all WebKit-based browsers on desktop, such as Apple Safari or Google Chrome. (Sencha Inc. Sencha Touch 2013 Demos)

3 ANALYSIS OF THE MOBILE SITE

In order to make the implementation more efficient, an analysis regarding design of whole site should be made in advance.

3.1 Design analysis

Design looks quite nice and clean, although there are quite many pages, the design keeps in same layout.

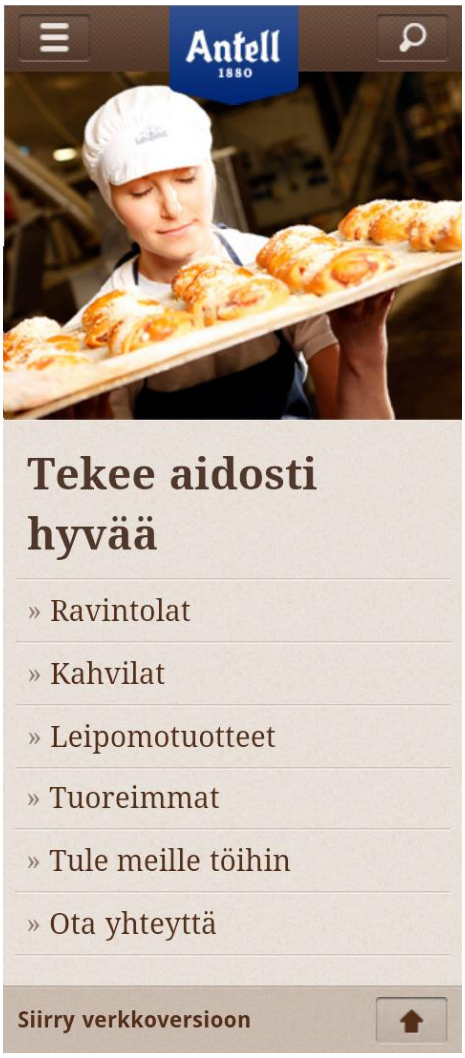


FIGURE 5. Antell mobile site front page screenshot

The general structure of the mobile site is as following:
Firstly a header appears which contains a sidebar toggle button at the left side, the company's logo in the center, and a search button at the right side. When the sidebar button is clicked, the main menu of the site is displayed. Similarly, if the search button is toggled, the search form will be shown.

Then there is the highlight picture; it is one of the key elements on the page which may attracts the user's attention, so it could be verified according to the content of the page. Under the highlight picture, the main content of the page is illustrated; usually it is a combination of textual and graphical description. The page ends with a footer section, from the design; there is one link to standard website and a "back-to-top" button.

As introduced earlier in the first sub-chapter of chapter one, the parts for general structure which is not need to always be changed should be added into template or include, at the same time, the main page content which should be editable for site manager will often be placed under individual page's "Sections" tab. In this case, the header and footer are the areas which always keep the same no matter how the page content changes, therefore, the header and footer should be implemented with includes. The reason for that is that in the future probably more templates are coming, for the possible more templates, there is no need to rewrite the code of header or footer, it would be very simple and time-saving by just adding the includes to those templates. Then the main content can be placed in template between header and footer includes.

3.2 Arrangement of the content

Usually there will be less content on mobile site than that on standard website; however it is highly important to keep the uniformity of the content on both mobile and standard sites. For that reason, it will be ideal if there is no need for site manager to add separate content for mobile site, the content for mobile site can somehow be copied from standard website. Holding on with that idea, the CMS will make it easy to do: since the page content will be added in the format of blocks on page section, and those blocks can be shared on both normal and mobile version of the same template. Thereby we just add the blocks which contains the content that need to be added to mobile site to the mobile template, place it where it should be. Then problem of content arrangement is solved!

3.3 JavaScript function

See from the preview images of screenshots of Antell's front page layout design that there are two sections clearly noticed which require JavaScript function. One is the header area, the menu button with three short lines and a search button with a magnifier icon. The idea for the menu button is that when that button is clicked, a main menu of whole site will be opened from left side, so that user can navigate to certain page they want no matter which page they are currently in. Similar for the search button, a search form will be opened if that button is clicked.

For that purpose, we need to use JavaScript for making the good effects or animation of opening and closing of the menu and search.

Then according to customer's requirement, they need such "Find closest café" and "Find closest restaurant" function. To make it more specific, the function should work in the following way: when user clicks "Find closest café" link, then the user's current geographic location should be detected, most mobile devices will automatically ask in advance if the user wants to share the current location. Then if the permission is give, the detected current location will be compared with stored address of all cafés, and those cafés which are in a range of three kilometers around the current location will be shown as closest cafés.

"Find closest café" and "Find closest restaurant" are basically same functions, thus I will just use "Find closest café" as example in the following content.

Divide "Find closest café" function into two parts technically: the first part is detect user's current geographic location and the second part will be filtering the cafés within a distance of three kilometers around current location.

Detection of current geographic location can be achieved by using HTML5 Geolocation, there is a method called "getCurrentPosition". Since the current location will be used for calculating distance among those stored addresses, so the textual address will not be proper, the current location should be transferred into format of latitude and longitude. “getCurrentPosition” method has properties for latitude and longitude. As an example, the JavaScript code in Appendix1 will first detect if your browser or device supports geolocation or not, then if it is supported, you will be asked to detect your current location by clicking a “Try It” button. The current location will be generated in latitude and longitude format.

4 IMPLEMENTATION

Implementation includes all necessary steps of actually coding for a website, from general structure build to very detailed CSS syntax and JavaScript function. It is the part of building a website which takes most of the time. However, if an efficient implementation is done, the time will be saved from following testing phase. All steps of implementation will be illustrated by turn according to the real processing.

4.1 Code for foundational page layout

As introduced in chapter 1, a structure of web page is defined from template. There are includes added inside template for the common part of different pages. The default template is used for desktop view; relevantly the mobile template will be taken into account when system detects a user loads website with a mobile device.

The important thing to note when making the template for mobile version is that content inside head tag in HTML document should be partly different. Head section is placed in an include with name of “HEAD_MOBILE”. For the purpose of a responsive view on different mobile devices, viewport meta tag like in the example beneath should be added into head section.

```
<meta name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1, maximum-scale=1, user-scalable=0">
```

The table 6 shows what each property does. (The table is originally from reference: The viewport meta tag. Date of retrievals 11.9.2013. Screenshot of the table on source site is too large to fit in the document, so a remade table is made with exactly the same content as that in original.)

Width inside content property of meta tag controls the size of the viewport. This value can be set to a specific number of pixels like width=340, or it could be device-width like in the former example which is the physical width of the device’s screen. In this case, CSS media queries will return the actual dimensions of the device, not the “zoomed-out” version, so that no scaling of the page occurs at all.

“The initial-scale property controls the zoom level when the page is first loaded. The maximum-scale, minimum-scale, and user-scalable properties control how users are allowed to zoom the page in or out (Sheppy, Vlad.V, martind1, samdutton, glztt, zigomir, Matt.B, wesj, 27.8.2013 Using the viewport meta tag to control layout on mobile browsers, date of retrievals 11.9.2013).”

TABLE 1. Attributes of content property of viewport meta tag

Width	The width of the virtual viewport of the device. Enter a number (pixels assumed), or the keyword "device-width" to set the viewport to the physical width of the device's screen.
Height	The height of the virtual viewport of the device. Enter a number (pixels assumed), or the keyword "device-height" to set the viewport to the physical height of the device's screen.
Initial-scale	The initial zoom of the webpage, where a value of 1.0 means no zoom.
Minimum-scale	The minimum level the user is able to zoom out of a webpage, where a value of 1.0 means the user isn't able to at all.
Maximum-scale	The maximum level the user is able to zoom in on a webpage, where a value of 1.0 means the user isn't able to at all.
User-scalable	Sets whether the user can zoom in and out of a webpage. Set to yes or no.

“To also prevent the device from zooming in on a webpage when its orientation has been changed from portrait to landscape and vice versa, you can also throw in an initial-scale and maximum-scale property and limit both of them to 1. After that, you can get to work with changing your page's layout depending on the device's dimensions, CSS pixel density, and more. Where to take things is up to you and your design sense! (The viewport meta tag. Date of retrievals 11.9.2013).”

Then still inside head section, a separate CSS file is added which is only for mobile view. That is because the desktop version and mobile version has different layout and page structure, code exclusively for mobile version will never be used for desktop, and thus there is no need to load that code when web page is read on desktop. In this way loading pressure is not increased so that the page can be normally loaded. In order to prevent CSS file for mobile be overwritten by that for desktop, it should be added after the file for desktop.

Where after, the body section should be filled with content that will be displayed on web page. Content usually comes from different blocks on pages from CMS admin. In the mobile template, main structure of body section is created based on the design. Some classes can be added to div tags exclusively for mobile version.

4.2 CSS specially for mobile version

After page is structured, CSS code should be added for showing website nicely on mobile devices. In order to separate from the CSS file which applies layout to desktop version, a new CSS file named mobile.css is created and attached inside header tag of HTML document.

Because there are many mobile devices with different screen sizes, it is difficult to set a fixed width or height for the elements on a web page to fit in all those different-sized screens. Therefore the fluid width and height should be used instead. In common, it is enough to set fluid width for most elements, and leave height automatically set. For images, if its width is set to be fluid, then the height will be resized automatically according to width. For the purpose of making width fluid, it should be defined in percentage in CSS file. Paddings and margins of the element could also be presented in percentage if necessary.

When coding for CSS file, it is a good habit to always keep writing some comments which shows what will this part of code do. In that way, if the file needs to be changed in the future, especially if somebody else than original author will make the change, it would offer much convenience for understanding the code.

For writing readable CSS code, it is very important to keep in good style which means code should be indented. Most of the modern web development IDE has the function of auto-indent which is handy to use.

Another tip for good CSS coding habit is to always test the page every time when some new code is added, particularly test it with Internet Explore which is known as the biggest challenge of browser compatibility. It will save huge amount of time for the final testing, because problems can be found and solved in time from previous testing.

Sometimes when working with a huge website, there could be thousands of lines in CSS file, the larger the file is, the more possibility of mistake there will be made. One way to help avoiding unnecessary spelling error is to use CSS shorthand. For example, defining padding for a div, the CSS code is:

```
div {
	padding-top: 10px;
	padding-right: 10px;
	padding-bottom: 10px;
	padding-left: 10px;
}
```

In shorthand it can be written as:

```
div {
	padding: 10px;
}
```

In CSS file of Antell site, shorthand is used as well.

Lots of similar shorthand can be used in CSS file, but those are not presented in thesis since that can be found easily by searching in Google.

4.3 JavaScript functions

From the analysis of the mobile site in chapter three, several JavaScript functions are needed to achieve different purposes. Not all the JavaScript functions used in Antell are introduced, but what will be explained are: sidebar sliding, scroll page back to top and find closest café.

4.3.1 Sidebar sliding

The idea for this function is to open and close the sidebar smoothly when sidebar button is toggled. In jQuery, there is a function called animate which can be used in this case. The sidebar and main content can be animated to slide to left or right, and the amount that element should be moved is the width of sidebar content. For example when opening the left sidebar, since the sidebar content is hidden earlier with a negative value of left position, the value should be changed to zero so that sidebar content will start from left side. At the same time, the main content area is moved to right side with same amount pixels, so that the opened sidebar and main content are placed side by side. Vice versa when closing the left sidebar, just set back the negative left position for sidebar content and zero value for main content area. Code for this function can be found from Appendix2.

4.3.2 Scroll page back to top

This function is very useful for mobile website since usually the screen of mobile devices is much smaller than that of desktops, therefore the page can be extended very long, with a back-to-top button at the bottom user can get back to top of the page quickly.

In jQuery animation function, there is scrollTop option which we can use to scroll the whole page to the top by simply set zero value for that option. Code for this function is:

```
$('.back-to-top').click(function() {
        if (isMobile.Android()) {
                window.scroll(0, 0);
        } else {
                $('html,body').animate({
                        scrollTop : 0
                }, 1000);
        }
                return false;
});
```

See from the code above, there is exception for Android devices, because jQuery animate does not work on some devices with low version Android and similarly some low version jQuery may also not working with some Android devices. Nowadays there are still few people using lower version Android than latest version, so that exception is necessary.

4.3.3 Find closest café

This function will work only if the browser has geolocation function and user allows to detect his/her current location. Check that first and if everything is ok then make the Google Maps initialization ready. In initialization, map should be defined and there are many available options for styling the map, like zoom level, center of the map, type of the map etc. Code example:

```
var mapContainer = $("#product-map");
var styles = [{ stylers : [{ hue : "#ffa200" }, { saturation : -20 }, { gamma : 0.85 }] }];
var map = new google.maps.Map(mapContainer[0], {
        zoom : 8,
        center : new google.maps.LatLng(64.453849, 28.575439),
        mapTypeId : google.maps.MapTypeId.ROADMAP,
        mapTypeControl : false,
        streetViewControl : false,
        styles : styles
});
```

More options can be viewed from Google Maps API reference official site (Google Inc. Google Maps Javascript API V3 Reference, date of retrievals 1.10.2013).

Once the user current location is detected, the latitude and longitude of it can be used for adding a marker to Google Maps. This marker should have different appearance than markers of found cafés, for example on Antell site, the marker of user's current location is in blue color whereas maker of café is in brown color.

Then go through all the café address and transit it into latitude and longitude use Google Geocoding (Google Inc. Google Geocoding API, date of retrievals 1.10.2013).

After that, Distance Matrix Service of Google Maps will be used for achieving the distance between user's current location and all cafés, then show those cafés which are within three kilometers of user's current location. Whereas if there are too many cafés, it may cause an error when using Distance Matrix Service, the error is "over_querry_limit". According to usage limits and requirements of Google Maps JavaScript API v3:

"The following usage limits are in place for the Distance Matrix service:

- Maximum of 25 origins or 25 destinations per request; and

- At most 100 elements (origins times destinations) per request.

Requests are also rate limited. If too many elements are requested within a certain time period, an OVER_QUERY_LIMIT response code will be returned." (Google Inc. Distance Matrix Service, date of retrievals 1.10.2013)

For each marker of café, there is info window in which the café name, address and a link to café page is shown. Use Info Windows (Google Inc. Google Maps Info Windows, date of retrievals 1.10.2013) of Google Maps for that.

5 TESTING

Testing for Antell mobile is done through group work in different versions of systems: Android, iOS, WindowsPhone. Mainly check for following aspects:

- Images and text are flexibly resized according to screen size and also in both portrait and landscape view.
- No broken layout, no overlapped element on all pages.
- No unavailable files or broken links or blank pages.
- JavaScript functions work nicely and the content is generated in same way after running JavaScript functions.
- No JavaScript error.
- Appearance looks the same on different devices. That mainly includes fonts, paddings, margins, colors, background, images, forms, and media.

It is better to test by different people than oneself, others can always notice some points which you may ignored.

Modern browser has very handy website checking tools for web developers. For example in Chrome and Firefox, just right click the element which you need to check for either HTML code or CSS, then choose “Inspect element”. In these development tools, CSS can even be modified and the changes will be applied immediately. Development tools is also useful for checking JavaScript errors. If there is any JavaScript errors, it will be shown in console tab of development tool and it shows even from which line in the file the error occurs. With the help of development tool, errors can be found easily and time is saved.

During testing, one problem of showing customized form elements is found. Originally the plan is to style the form elements like dropdown list, checkboxes and radio buttons with customized design, however that causes problem on some mobile devices. That is because it is not possible to directly change the CSS layout for input, there is always some default styles which cannot be removed.

However if keep the default CSS style and add extra tag like span wrap the input, and add the style to the span instead of input. This could work but new problem appears, for example the checkbox, the alignment of checkbox icon and text is difficult to set horizontally, it may work on some devices, but may be broken on other devices. Also different devices have different default padding and margin for input, so it is even harder to make the appearance look~~s~~ universal on all devices. Consider of all those reasons, decision is made to not use customized layout for form element but just leave that with default style. Even if the default style of form elements are varied on different devices, as long as it looks good and is not broken, it will make better user interface experience than a cool designed elements but with messy layout.

6 CONCLUSIONS

Author of the thesis is responsible for the whole implementation of Antell's mobile site, which includes building of template, include and modules on CMS, coding for all CSS files and JavaScript files.

Processing of the implementation briefly contains following parts: Create general structure which means templates on CMS in this case. Fill out the pages with content, which can be done by adding different modules to CMS. Style the page in CSS files. Make page functional by adding JavaScript functions.

After the initial implementation is finished, the mobile site is tested by a group of experienced web developers. The test is made on all popular mobile operating systems. Bugs are reported after testing, those bugs should be fixed one by one. After that the mobile site is ready to be published after client finish adding content to website.

Both the desktop and mobile version of Antell site are published now. No major error is reported till now. So far so good.

When writing the bachelor's thesis, lots of research~~ing~~ work has been done for retrieving related information, through that the processing of making a mobile website is reviewed and emphasized. Moreover, much of outspread knowledge is learnt as well at the same time, which helps the author go further in the development of mobile website. For example even though on case mobile website, there is no framework used, thanks to the chapter of general information of mobile development, some frameworks need to be introduced, so author studies about those frameworks and gets to know the features. That will be useful when picking up a mobile framework for some projects in the future.

On the whole, this thesis not only contains all steps of how a mobile website can be produced in general, but it also helps author of the thesis review the case project, examine if there is any file needs to be updated, check if any part of the project can be improved with latest technologies

7 DISCUSSION

In this thesis, a common processing of making a mobile website is introduced with a real-life case. Among the content of thesis, the comparison between desktop and mobile version is mentioned for many times from different aspects. Web development for mobile devices has become popular only in recent years with the emergence of smart phones. Thus the comparison can help to understand better of how to get it work on mobile devices with the knowledge of desktop version development.

Case website is implemented on a CMS called Sivuviidakko. Sivuviidakko is introduced only in general but after all, use CMS for making website is getting more and more popular today and there is various CMS software available already. Topic of CMS research is suggested which seems to be interesting.

Although this thesis is about making a mobile site, it is not the most popular way of showing website on mobile devices any more. Instead, responsive website is used more and more. For making only one version of website, just use media query and define CSS for different sized screens, then the website can be responsively resized and fit the screen on both desktop and mobile devices. For that reason, another recommended topic is development of responsive website.

REFERENCES

Google Inc. Distance Matrix Service. Date of retrievals 1.10.2013
https://developers.google.com/maps/documentation/javascript/distancematrix

Google Inc. Google Geocoding API. Date of retrievals 1.10.2013
https://developers.google.com/maps/documentation/geocoding/

Google Inc. Google Maps Info Windows. Date of retrievals 1.10.2013
https://developers.google.com/maps/documentation/javascript/examples/infowindow-simple

Google Inc. Google Maps JavaScript API V3 Reference. Date of retrievals 1.10.2013
https://developers.google.com/maps/documentation/javascript/reference

Gube.J. 2011. Top 10 Mobile Web Development JavaScript Frameworks. Date of retrievals 20.4.2013
http://sixrevisions.com/javascript/mobile%C2%A0web-development-frameworks/

jQuery Foundation. jQuery Mobile 1.3.2 Date of retrievals 9.9.2013
http://view.jquerymobile.com/1.3.2/dist/demos/intro/

Lanese.L. 2012. What can a Mobile Framework do for you? Date of retrievals 19.4.2013
http://leolanese.com/blog/2012/08/03/what-can-a-mobile-framework-do-for-you/

Liana technologies. 2012. Date of retrievals 18.3.2013
http://www.lianatech.com/company/liana-technologies.html
http://www.lianatech.com/products/lianacms.html

Markus.F. Mobile Frameworks Comparison Chart. Date of retrievals 19.4.2013
http://www.markus-falk.com/mobile-frameworks-comparison-chart/

Raasch.J. 2010. How to build a mobile website. Date of retrievals 18.3.2013
http://mobile.smashingmagazine.com/2010/11/03/how-to-build-a-mobile-website/#mobile-stylesheets

RIMMEL UK mobile site. Date of retrievals 19.3.2013
http://uk.rimmellondon.com/mobile

Sencha Inc. Sencha Touch 2013 Demos
http://www.sencha.com/products/touch/demos/

Sencha Inc. Sencha Touch 2013 Features. Date of retrievals 9.9.2013
http://www.sencha.com/products/touch/features/

Sencha Inc. Sencha Touch 2013 Who’s using? Date of retrievals 9.9.2013
http://www.sencha.com/products/touch/whos-using

Sheppy, Vlad.V, martind1, samdutton, glztt, zigomir, Matt.B, wesj 27.8.2013 Using the viewport meta tag to control layout on mobile browsers. Date of retrievals 11.9.2013
https://developer.mozilla.org/en-US/docs/Mozilla/Mobile/Viewport_meta_tag

The viewport meta tag. Date of retrievals 11.9.2013
http://www.javascriptkit.com/dhtmltutors/cssmediaqueries3.shtml

DETECT LOCATION CODE

APPENDIX 1

```
<!DOCTYPE html>
<html>
    <body>
        <p id="demo">Click the button to get your coordinates:</p>
        <button onclick="getLocation()">Try It</button>
        <script>
            var x=document.getElementById("demo");
            function getLocation()
            {
                if (navigator.geolocation)
                {
                                    navigator.geolocation.getCurrentPosition(showPosition);
                }
                else{
                            x.innerHTML= "Geolocation is not supported by this browser.";
                }
            }
            function showPosition(position)
            {
                x.innerHTML= "Latitude: " + position.coords.latitude + "<br>Longitude: " +
    position.coords.longitude;
            }
        </script>
    </body>
</html>
```

SIDEBAR SLIDING FUNCTION

```
$('.header .sidebar').click(function() {
        var check = $(this).parents('body').find('.sidebar-menu').hasClass('open');
        if (check) {
                $(this).parents('body').find('.sidebar-menu').removeClass('open').animate({
                        left : '-254px'
                }, 300);
                $(this).parents('body').find('.wrap').removeClass('open-sidebar').animate({
                        paddingLeft : '0'
                }, 300);
        } else {
                $(this).parents('body').find('.searchform').removeClass('open').hide()
                .css('left', '-254px');
                $(this).parents('body').find('.wrap').removeClass('open-search').animate({
                        paddingLeft : '0'
                }, 0);
                var h = parseInt($('body').height());
                var h_screen = parseInt($(window).height());
                if (h >= h_screen) {
                        $(this).parents('body').find('.sidebar-menu').addClass('open')
                        .css('height', h).show().animate({
                                left : '0'
                        }, 300);
                } else {
                        $(this).parents('body').find('.sidebar-menu').addClass('open')
                        .css('height', h_screen).show().animate({
                                left : '0'
                        }, 300);

                }
                $(this).parents('body').find('.wrap').animate({
                        paddingLeft : '254px'
                }, 300, function() {
                        $(this).parents('body').find('.wrap').addClass('open-sidebar');
                });
        }
});
```