Examensarbete

Web Design for Multiple Platforms in Microsoft SharePoint 2013

av

Erik Claug

LIU-IDA/LITH-EX-A--13/072--SE

2013-12-20
Examensarbete

Web Design for Multiple Platforms in Microsoft SharePoint 2013

av

Erik Claug

LIU-IDA/LITH-EX-A--13/072--SE

2013-12-20

Handledare: Anders Fröberg
Examinator: Erik Berglund
Abstract

As mobile devices become more common when surfing the web there is an increasing demand of making web sites adjusted to other devices than computers. One common platform on the web is Microsoft SharePoint. In this report I look into one method of how to develop web sites that work on multiple devices in Microsoft SharePoint. I will be looking into different techniques as separate web sites compared to responsive web design. By exploring the approach of mobile first I try to find a working solution of techniques. I also look into another method that is suggested in a book and look into how Bootstrap could be used in the process.

I will also look into content to identify what content need to be adjusted between different devices. I find that navigation is an important part of a web site and look into different methods of handling navigation on mobile devices.

By analyzing different techniques and looking into what content is going to be adjusted a solution is proposed where the navigation is adjusted in a responsive design. The design is converted into a SharePoint masterpage and then fixed using stylesheets and JavaScript. The solution works on a web site with a complex site structure since it focuses on how to adjust the navigation.
## Contents

1 Introduction ........................................... 2  
  1.1 HOW Solutions .................................... 2  
  1.2 SharePoint ......................................... 2  
  1.3 Mobile web ......................................... 3  
  1.4 Goal .................................................. 3  
  1.5 Problem ............................................ 3  
  1.6 Scope and limitations ............................... 4  
  1.7 Method .............................................. 4  
    1.7.1 Case Study Research ......................... 4  
    1.7.2 My interpretation ............................ 5  
  1.8 Sources of information ............................. 6  
  1.9 Structure .......................................... 6  

2 Data Collection ........................................ 7  
  2.1 Observations ....................................... 7  
    2.1.1 Content ....................................... 7  
    2.1.2 Navigation .................................... 8  
  2.2 Techniques ......................................... 9  
    2.2.1 Separate Web sites ............................. 9  
    2.2.2 Mobile First .................................. 9  
    2.2.3 Responsive Design ............................ 10  
    2.2.4 RESS ........................................... 11  
  2.3 Summary ........................................... 11  
  2.4 Related work ...................................... 12  

3 Approach .............................................. 13  
  3.1 Setting up the project ............................. 13  
  3.2 Creating new files .................................. 13  
  3.3 Navigation ........................................ 14  
  3.4 Working with stylesheets and JavaScript ........ 15  
  3.5 Scaling up ......................................... 17  
  3.6 Converting ......................................... 18  
  3.7 Repairing .......................................... 19  
  3.8 Result ............................................. 19
## 4 Discussion

4.1 Influence of the method ........................................ 21
4.2 Bootstrap .......................................................... 21
4.3 RESS ............................................................... 22
4.4 Shadow DOM ....................................................... 22
4.5 Future work ......................................................... 23

## 5 Conclusion ......................................................... 24
List of Figures

3.1 The very simple HTML: ........................................... 14
3.2 Example of how Adlibris.com and iPhone settings handles navigation: .......................... 15
3.3 One of my sketches on how to design the navigation with the desired behavior: ................... 16
3.4 The finished implementation of the mobile menu: ............ 17
3.5 The finished implementation after scaling up to desktop: ... 18
3.6 The finished implementation in SharePoint after repairing: .... 20
Chapter 1

Introduction

When developing web sites for many different devices some adjustments need to be done. There are also a lot of different techniques that can be used when doing this. As this is part of the front end, how can this be done when working with a back end that you have limited control or knowledge of. I will look into how to make web sites work on different devices when working with Microsoft SharePoint 2013 as back end. This will be done with very little knowledge in SharePoint.

To understand why this is done I need to know some about the background. I will look into what the company as a stakeholder wants. I also look a little bit into what SharePoint is and also why the mobile web is so important.

1.1 HOW Solutions

HOW Solutions is a company that works with different IT solutions. One of the solutions they have specialized in is Microsoft SharePoint. A lot of their customers request their desktop solutions to work on mobile as well. Therefore they want to get a recommendation on how to make a mobile solution in SharePoint.

1.2 SharePoint

Microsoft SharePoint is a platform of web applications with a big focus on intranet. It contains a lot of tools for collaboration, content management and document management.

The system is integrated with a lot of other Microsoft products which makes it an easy choice for many companies that already use Microsoft Office
for instance. Nowadays SharePoint also contain functionality of publishing web sites and this is a feature that companies want to use, since they already paid for the license. SharePoint also introduced a way to handle mobile web sites in their last version where they let you assign different channels for different devices. These channels then renders different content. One goal is to see if this is the way to go. The worldwide marketshare for SharePoint in public web sites is 0.1%\(^1\) and in Sweden it is 0.3%\(^2\).

Developing in SharePoint is quite advanced and some people I talked to at HOW Solutions claim that it takes at least a year before you can do any development if you haven’t worked with it before.

### 1.3 Mobile web

Surfing the web is moving from desktop computers towards mobile devices such as smartphones and tablets. Today around 65% of the people in Sweden access the internet using mobile phones\(^3\). This makes companies more aware of having web sites that are adjusted to mobile devices. It also makes companies that already use SharePoint have a demand of making that content adjusted for mobile devices.

### 1.4 Goal

The goal of this work is to find a method to make a web site created in SharePoint adjust to the mobile devices that visit it. The goal is to both find what technique and method could be use to achieve this. I will also investigate what parts need to be adjusted.

### 1.5 Problem

The work is based on the following main problem:

- How can a web site in SharePoint be implemented to fit mobile devices by a person who knows nothing/little about SharePoint?

This can however be broken down into sub problems:

- How do you implement a web site to fit multiple platforms?

- Is there a way to implement web sites in SharePoint without knowing that much about SharePoint?

---


1.6 Scope and limitations

The aim of the study is to find what contents need to be adjusted and what techniques to use. It does not aim to investigate the usability or optimize performance.

The work is limited to 20 weeks which means a solution cannot require too much knowledge of SharePoint since it requires quite a lot of time to learn. This also means that only one solution will be evaluated.

The requirements of the solution is based on the customers of HOW Solutions which are mainly different municipalities. The customer’s required functionality is what the solution should try to achieve.

The work will only involve the publishing part of SharePoint and no other parts such as wiki or collaboration rooms.

The solution is limited to the latest standards to see what can be achieved with the latest technique. This means that the solution might not work in old web browsers or old mobile devices.

The work is limited to how input and layout differs between different devices. It will not look into how connectivity differs between them.

1.7 Method

First the idea was to make an experiment with a few different solutions and compare them. One of them would be using device channels with a mobile version and a desktop version. However when making the mobile version I realized that making it responsive would take less time than making a desktop version too. I also realized that I would not be able to learn enough about SharePoint in this limited time to create a version based on as many techniques from SharePoint as possible. At that time I started to realize that there was almost just one way I could do this. Therefore this led me more into doing a case study rather than an experiment to investigate that method more.

1.7.1 Case Study Research

When doing a case study you create a more specific case that you study. It doesn’t require any strict boundaries between the thing you study and its surroundings. It also gives you a deeper understanding of the thing you study [1] [2]. There are a lot of guidelines you can use when doing a case study. The guidelines involve:
1.7. METHOD

1.7.1 How to design the case? This includes what the objectives are and finding research questions.

• How to collect the data, including classification of sources, how to do interviews or focus groups and do observations.

• How to keep protocols and do the analysis of the data.

• How to report the findings.

The theory of case study in software engineering aims to make case study in this field more structured. Therefore my interpretation is that I can use the recommendations from it that suits me and decide on my own how strict I want to be about the theories.

1.7.2 My interpretation

When I did my case study I used my goal and problems as starting points. From this I stated my case as:

• Create a web site for multiple platforms designed for a municipality hosted on Microsoft SharePoint 2013.

The objectives were extracted from this as follows:

• What needs to be adjusted?

• How is a web site adjusted to work on multiple platforms?

• How can I get this functionality in SharePoint?

The next step was to start answering these questions. This was done by both observing existing solutions and by looking at theories about good solutions. This brought up solutions that required new iterations of research. When doing observations I kept a simple protocol of the findings. When looking at different theories I just stated the advantages and disadvantages of them.

When my research was done I stated a theoretical solution on how to solve this specific case. I also proved this solution to be working, this can be found as my result in the Approach chapter. Finally I looked into other solutions and future work, this will be brought up in the Discussion chapter.

During my implementation I also used some theories of computer engineering. I used iterative and incremental development where I started of by just implementing the content, and then throughout the iterations modified and refined it with more functionality.
1.8 Sources of information

In this work the sources will be of varied kind. Since the development of the web is going very fast, both because of the techniques being used but also because the platforms change all the time, official recommendations might not be up to date. When looking at these the date of publishing have to be considered. This means that a recommendation of a developer might be more trustworthy. When using blogposts and similar I have to be critical of the sources and justify their trustworthiness by comments and how much they have been shared.

1.9 Structure

This first section covered the method I will be using and also stated how my case is designed. The following chapter will be a data collection where I research the relevant parts of this case. That chapter will end with a theoretical solution. The next chapter is a walkthrough using that theory which aims to prove the theory to be working. After that follows an analysis of the process and of the result. Finally there will be a discussion where I look a little bit outside the scope and into the future.
Chapter 2

Data Collection

The process of collection data will be iterative as the method describes. I will start researching information and that will raise other questions. Some of the information presented here will be based on observations and some information will be theories that others recommend. After these findings are presented they will be combined into a theoretical solution. The observations are reported and can be found in the appendices.

2.1 Observations

To be able to design a web site for a municipality I will need to see what is the most common content on current sites. I also need to look into which ones of these parts need to be adjusted and then how this adjustment could be done. All of this will be done through observations.

2.1.1 Content

When looking at some different municipality web sites I can easily see that they are very informative sites. They contain a lot of text and images and they are spread over a lot of different pages. The text is good at adjusting itself to different screen sizes. Images could need more adjustment between different devices, but it is more related to performance which I will not cover in this report.

When having the content spread over a lot of pages it leads to a complex navigation structure. When observing three different municipality sites I can see that all of them have a site structure that is at least 4 levels deep. All of them have 6 or more options in the top level. One of them has as many as 9. All of them often have around 10 options in the sub levels.
Another thing I observe is that all of the sites that have sub sites contain information. Therefore I seem to have a requirement that all links to a site should make me go to that site, even if there are sub sites to it. I will come back to this issue later.

To sum it up, navigation seems important in this kind of web site considering the complexity of the site structure and is something I need to focus more on. This will be investigated more thoroughly in the next iteration.

2.1.2 Navigation

This was brought up during the second iteration since I found out that navigation was very important for this kind of web site in the first one. Now I need to find out more about how others handle navigation when going mobile.

The navigation or menu of a web site is commonly a horizontal bar at the top of the site. That is the case in the sites I observed too. This is a design that works fine on desktop computers that have a screen in landscape mode. But how does this work on a small mobile device? This kind of menu is also commonly completed by a left navigation that contains a sub menu for the current site. This would take up all the space on a mobile device, therefore that is not a solution.

To find a solution on how to adjust this to a smaller screen I will observe some solutions. Sony.com have many different choices in the top menu, many in the sub menu and three levels deep. They solve the problem by stacking the top menu items on each other. The menu is shown by clicking a button. Each sub menu slides in from the side when the level above is clicked.

Looking at the iPhone settings menu each sub menu slides in from the side in the same way. It also has a lot of different choices at each level and is quite many levels deep.

Adlibris.com also stacks the menu items on top of each other and slides in the menu from the side. The sub menus are listed on each sub site and cannot be found in the top navigation. This means that a person need to load a page to see the sub menu of it.

A common solution is to let the sub menu slide in from a side when browsing through the navigation. Adlibris wants their customers to be able to look at a site even if it has a sub menu, just as I want to as I found out in the last section. Adlibris handle this by loading the site and show the sub
menu separately on it. This requires a lot of reloads when browsing through the navigation.

2.2 Techniques

The observations was important to realize what to focus on in my specific case. However another important part to examine is what technique to use when implementing this. To find that information I start researching the different techniques there are and write down the most relevant information about them.

2.2.1 Separate Web sites

A technique that has been used a long time is to design separate versions of the same web site for different platforms. One for desktop computers and one for mobile devices. In that way the traditional desktop site could be kept as it was and a mobile site could be added. This was the natural way to do it when mobile devices started to support web browsing. To do this the server detects what device is accessing the site and then serves the client with the appropriate one. In many cases the mobile device is sent to m.example.com or mobile.example.com. This can make links and bookmarks work very bad or not at all. The server often determines this information by using the user agent string that the client sends to the server. This requires that the server interprets the string correctly. If it doesn’t the client might get the wrong site. This concept requires that two sites are developed and maintained. It also requires that the list of user agent strings is maintained. The biggest advantage of this technique is that since the server renders different content it can serve mobile devices with less data.

2.2.2 Mobile First

This is not as much a technique as it is a way of thinking. It is a procedure used when developing web sites for multiple platforms. It states that you develop the site with the mobile view as a starting-point. Since the smallest screens are most limited you start there and make sure that the required functionality works as intended despite the limited screen. When this is achieved you move up to the next screen size and adjust the functionality to the slightly less limited screen. This is done until you build the site for the biggest possible screen you want to support.

This is a kind of progressive enhancement. This concept was used already ten years ago when different web browsers had different support. The idea was to make sure that the browser with least compatibility should support the page and functionality was added gradually for web browsers with better support. In that way a web site was built that worked in older
browsers at the same time as it used functionality from newer browsers. In the same way mobile first makes sure that the site functions on the smallest screen at the same time as it adds even more functionality as the screen grows.

The opposite of progressive enhancement is called graceful degradation. This means that you would build the web site for the most compatible web browsers first and then remove functionality when using browsers with lacking support. There is no corresponding term when talking about mobile first, but it would mean that you first build the web site for the biggest screen and then change the site when smaller screens use it.

Building sites for desktop computers and scaling them down is the method that has been used most frequently before since the web was first designed for desktop computers. Mobile devices has started to access the web more recent. The problem when using this procedure is when you build the site exactly as you want it and with a lot of space for it, it is hard to know what to remove and how to scale it down when moving to smaller devices. There is also a big risk that the focus moves from functionality to design.

Mobile first is something that is recommended by many and it states that when developing for the mobile there will be more focus on the core of the web site which will make the site more functional. When you have a functional web site that works for the mobile device it is easier to add content for a bigger screen, than it is to remove content when going towards a smaller screen. Many source state stat this makes more usable mobile sites but also makes the web site more functional on all platforms.

### 2.2.3 Responsive Design

Responsive web design is about letting the client decide how the web site will be shown. This is controlled by the width of the screen. This means that you don’t care about what device it is, just the size of it. When designing the layout you set breakpoints where you think the screen is wide enough to contain more content. Between the breakpoints the design can be kept to constant sizes or it can use relative sizes. When using relative sizes there will be more of a smooth change of the design between the different sizes.

To use responsive design the screen must be measured in some way. This can be done in multiple ways and JavaScript has been a common method. However today the most common way is to use media queries in CSS. You simply specify at what screen widths different CSS-rules apply. When combining this with mobile first you most commonly only use the minimum with as a rule within the stylesheet. This means that you build the general design for the smallest screen and when a screen size is at least the size specified
in the media query you add content. There could be an infinite number of breakpoints, but commonly they are set at a size where the design requires a change, or at sizes of known devices.

The big benefit when using responsive design is that you only build one web site to cover all possible devices that is going to visit it. The web site adjusts itself based on the client visiting and will always look the best way possible. This means only one site needs to be built and only one site needs to be maintained. This also means that all links and all bookmarks will work as intended since there are no different location of the site for different devices. In a world where sharing links is very common this is a big plus. This technique doesn’t require device detection. Responsive web design is the technology that Google recommends.

2.2.4 RESS

RESS is responsive design with server side components. This means that a lot of the advantages with responsive design is kept at the same time as some components can be rendered different at the server. This is a good solution when optimizing performance. However since I am not looking into the performance, this solution is not going to help me.

2.3 Summary

Responsive design seems to be the big recommendation around the web which makes it the obvious choice for my implementation. It also brings a lot of benefits like: only one site needs to be built, it will fit all devices and links will always work as intended. Designing with mobile-first in mind will make me think more of the content and the functionality in first hand.

When it comes to what content to work with I will definitely have to work on the navigation. It is really a part that needs to be adjusted depending on the screen size. It is also a part that really is important in this case.

The navigation should be both wide and deep in the sense of menu items. It is also a requirement that each menu item should lead me to a page even if it has a sub menu.

Now I need a method to get this to work in SharePoint. The next section will present a method.
2.4 Related work

The book Pro SharePoint 2013 Branding and Responsive Web Development present a procedure where they create a responsive web design in SharePoint. I have decided on going with a responsive design so this can probably be used by me too. In this book they start of by designing a regular web site, the desktop way. Then they use a new feature in SharePoint 2013 to convert it into a SharePoint master page. After that they use Twitter Bootstrap to make the site responsive and work on mobile. Finally they state that they have to fix the site since there are conflicts between the stylesheet from SharePoint and the one from Bootstrap.

This seems like a way to accomplish what I want to do. I will use my theoretical solution instead of Bootstrap together with their method to achieve this in SharePoint. The next chapter will present that procedure.
Chapter 3

Approach

In this chapter the procedure of the implementation will be described. The different steps taken will be described with details about different possible solutions and motivations of why one decision was taken over another. Problems that occurred will also be brought up and some nice pictures will illustrate what happens through the process.

3.1 Setting up the project

The first step is to have a SharePoint project to brand. This could be an already existing project. The more content the site has the easier it is to test things later. This process could be done by a SharePoint developer or administrator since it requires a lot more knowledge of SharePoint than the rest of the process does. One thing I did discover is that the different kinds of navigation render different HTML. The one called managed navigation is the one that generates the HTML I want, so this is more or less a requirement. However this kind of navigation has a lot of other benefits so it should not be too big of a problem to change it.

3.2 Creating new files

Now I can forget about SharePoint for a while. At this point I am just going to work with HTML, CSS and JavaScript. However since I will not be able to build the HTML of the navigation it is best to copy and paste the navigation SharePoint generates for me. This is easily done by looking at the source code of the existing SharePoint project. I also need to keep in mind that this part of the HTML I cannot change.

So I create the HTML with the navigation pasted. I create a CSS for my stylesheet and a JavaScript-file because it will be needed for some features.
3.3. NAVIGATION

CHAPTER 3. APPROACH

The HTML-file is implemented using the HTML5 standard and it should be as simple as possible. There shouldn’t be any inline-styles and for this simple design I don’t even need any class names and very few IDs.

```html
<html>
<head>
    <link href="css/style.css" type="text/css" rel="stylesheet"/>
    <script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>
    <script src="js/menu.js"></script>
    <meta name="viewport" content="width=device-width, user-scalable=no">
</head>
<body>
    <div id="inner">
        <header>
            <div id="logo-container">
                <img id="logo" src="img/logo.png"/>
            </div>
        </header>
        <nav>
            <section>
                <footer>
                    <div>
                        </div>
                    </div>
                </footer>
            </section>
        </nav>
    </div>
</body>
</html>
```

Figure 3.1: The very simple HTML.

3.3 Navigation

Before I start to build everything using CSS and JavaScript I need to know what to build. The first requirement of the navigation is that it should fit the smaller devices’ screen. The second requirement of the navigation is that I should be able to browse through the whole structure of menus and sub menus without the need to visit the pages they link to. The third requirement is that every single page represented in the navigation have to be possible to visit. This should work even if the item has a sub menu. These requirements are what I found when doing the observations.

To see how this could be implemented I look into a lot of different existing solutions of navigation. From what I found out I want the navigation to be hidden and shown when clicking a button. Adlibris uses this button as seen in the figure.

There seem to be two major ways of displaying the menu. One is that it shows up below the button and one is that the content of the page slides in one direction making the navigation appear. Adlibris’s shows up from
3.4 Working with stylesheets and JavaScript

Now that I have an idea of what all of my requirements are it is time to implement these. Starting with the stylesheet in CSS all the layout of the

Figure 3.2: Example of how Adlibris.com and iPhone settings handles navigation.

the side by sliding the content away, while Sony’s just appears below the button.

I will be using the simplest one since this is about making functionality and not the best usability. It doesn’t need to be more complicated than to give the right functionality and just showing the menu seems like the simpler way. When browsing through the menu there are two major ways of showing sub menus. One is to let the sub menu appear below its parent, like Adlibris does it, and the other is to let the sub menu slide in from either side, like iPhone does it. The solution where it appears below is simpler, however it will take a lot of vertical space as soon as I get a lot of levels in the menu. Therefore I will implement it sliding in from one side.

The last problem is how to both browse through the menu but also be able to visit each item. This will be solved by adding an extra button to the menu items that have a sub menu. Clicking this will show the sub menu, clicking the item itself will visit the page. I also have to make sure you can go back from a sub menu. This is fixed with a back-button.
menu is done. Since I go mobile-first this is just done with the small screen in mind. The menu objects are stacked on top of each other. All sub menus are positioned to the right of their parent. Some styling is also added to make sure each item is big enough to be clicked by a big finger on a touchscreen. Also, no hovering functions are implemented at all since these are not handled very well by touchscreen. The first click is classed as hover and the second as a click, making everything appear very slow. All of the CSS I add use hierarchal selectors to find the right object. In that way I don’t need to worry about using classes or IDs where I couldn’t add them anyway.

Now it is time to handle the hiding of the menu. This is easily done with CSS using the display: none; attribute. But how do I apply this to the menu. This has to be done with JavaScript since there are no good ways of doing it just using CSS. The JavaScript (using jQuery for convenience) listens to clicks on the button and adds or removes the hidden-class from
the menu.

Adding the arrow to the sub menu and the back-button would be easy if I had control over the HTML. Now I don’t so the JavaScript will help me with that. It simply looks for all menu-items with children and adds an extra link on those items. Styled it looks like a separate button. The same is done with the back-button.

![The finished implementation of the mobile menu.](image)

Finally I use JavaScript to look for clicks on these buttons and add or remove classes to show or hide the sub menus.

### 3.5 Scaling up

The next step is a bit of an iterative step. I will now decide what size I want to add a breakpoint to. I decide that the next size I want to have a different look is when using a tablet in portrait mode. It would have a width of 768 pixels. I add some new rules in the stylesheet when the screen width is at least 768 pixels. The only thing I decide here is to set a fixed width to the menu instead of letting it use all the width. However a lot of different
attributes could be added to the stylesheet at this point to add content to the bigger screen. I add another breakpoint which would represent a tablet in landscape. I set this point at 1024 pixels. The new rule here is to let the menu become a full horizontal top menu. This is quite easily done by adding some rules to the CSS when the width of the screen is at least 1024 pixels. In this width the sub menus will not show as before either as the layout now has room for a left navigation containing them. As I said earlier, this is an iterative process where you decide on a breakpoint and then write the CSS to show it the way you want.

Figure 3.5: The finished implementation after scaling up to desktop.

3.6 Converting

The next step requires some knowledge of SharePoint. I simply log in as a user with permission to change the looks of the site. All of my files need to be uploaded to SharePoint, which is simply done mapping the SharePoint master gallery as a network device. When this is done, using designmanager I go to edit master pages where I convert and publish the HTML-file I just uploaded. SharePoint requires that the file is semantically correct, but as long as I followed HTML5 I shouldn’t have any problems. Now I can either preview the page in design manager or I can apply this master page to the design and view it in action. In the preview mode I find code snippets, and this is how I integrate the real SharePoint menu. I pick the top navigation snippet, and get a lot of options to fix. The most important is to ask SharePoint to show as many static levels of the menu as I have. When all options are set I click update and I get all code I need in a textbox. I cut this and paste it to my HTML file on the mapped network drive. As I can see SharePoint has added a lot of code to my HTML file. But there is nothing
to worry about as long as I replace the hardcoded navigation I used when designing with the newly generated code. Now I shouldn’t need to worry about the HTML any more.

3.7 Repairing

Since I made a lot of design in my CSS and SharePoint have their own CSS I get some conflicts between them. SharePoint select their elements by ID or class and I used hierarchal selectors. SharePoint’s way is more specific and thenoverrides ours. I couldn’t really use classes or IDs since I didn’t write the HTML. (I could probably use some of the ones SharePoint use, but there are many.) The easiest way to fix these conflicts is basically to let my JavaScript remove all classes within the nav tags. In that way SharePoint’s CSS will not affect the navigation at all. After doing this using the developer tools in any browser you can find what parts doesn’t look like they should and what causes it. Then I just try to override it using either CSS or JavaScript. I do this until I get the desired looks and behaviors.

3.8 Result

The implementation required quite a lot of steps but it resulted in a working solution. The navigation, which was the most important component, worked with the data SharePoint gave us. It works perfect on a mobile device and it scales up on bigger devices. The process was most about writing CSS which is something a lot of people know and is quite easy to learn. The implementation resulted in some conflicts that I manually had to remove, but future work could help me reduce this problem.
Figure 3.6: The finished implementation in SharePoint after repairing.
Chapter 4

Discussion

The result of my implementation is a true responsive web site that will work in this specific case. The method I used could work on other content as well, however an analysis of the content would then be necessary. Some other things could also be interesting to look at in the future. Let’s look into some of them.

4.1 Influence of the method

Since my method only looks at a specific content of a web site, the result only becomes relevant if this kind of content ever is used in SharePoint. As my case is very simplified, the easiest way to present it could be a static web site that is not based on a platform like SharePoint. Therefore it is hard to say if SharePoint ever would be used for the content I examined in my method.

In this method I also looked into a very narrow part of SharePoint. This part is the publishing part which only handles the publishing of web sites. Dynamic content and more advanced features require other parts of SharePoint and these parts have not been tested. Therefore the result doesn’t really apply to the real case.

4.2 Bootstrap

The solution presented by [3] uses Bootstrap to get a responsive solution. This solution would probably work in many cases, however in my case the big focus was the navigation, and the navigation Bootstrap presents would not meet the requirements based on the observations. The solution they present doesn’t use the latest version of Bootstrap either which then doesn’t provide a fully responsive solution out of the box. Finally I already got quite a lot
of conflicts between my stylesheet and SharePoint’s stylesheet, introducing one more would give me even more conflicts to handle. In this specific case Bootstrap would give me more work to do than the advantages it provides.

4.3 RESS

Earlier I mentioned RESS as a solution that still gives a true responsive web site but also offers performance advantages by rendering components on the server. This could probably be one of the advantages when using SharePoint, since it has some built in support for it. In this specific case it wouldn’t have helped that much since the navigation would be using the same HTML on mobile as on desktop. However for other components it could be used to render different components for the different platforms to optimize performance. It still relies on user agents which means this will have to be handled correctly. Since this case didn’t have any advantage of different rendering I didn’t look any more into SharePoint’s support of this, but it would absolutely be a recommendation when working with performance related components.

4.4 Shadow DOM

The shadow DOM is a part of the document object model (DOM) that isn’t traversed but still renders. You can think of it as the parts of a widget, where the widget can be traversed, but not the components of it, even though they are rendered. These components are not affected by the CSS connected to the page. This could possibly be used when working with SharePoint to come around conflicts. By creating widgets in this way you wouldn’t have to worry about CSS conflicts. This is one thing that could be explored some more in the future.

Polymer is a front end framework that among many techniques use the shadow DOM to provide a platform for developing web sites. The aim is to encapsulate code and make the web more component based. Examining this platform could probably also give some help when developing for an existing platform with the risk of conflicts.

There are probably a lot more ways to achieve a more component based development of the web. This is absolutely technology that could be examined in the future to see if there is something to bring in with my solution to this problem.
4.5 Future work

In the future there will probably be a greater separation of front-end and back-end. The back-end would contain data structures while the front-end would contain the design and present the content represented in the data structures. This could be done using REST APIs which give a simple interface for the front-end to communicate with the back-end. This would let front-end developers and back-end developers focus on different things. Front-end developers would be the ones working with design, usability and also the ones making the site responsive.

As SharePoint has support for REST API it would be interesting to investigate how this could be used to make a responsive design. This could then be built with HTML that will not be converted to a SharePoint master page and would not get the problem with conflicts of stylesheets as I did. I could also easily build the navigation I wanted without being dependent of how SharePoint creates it for me.
Chapter 5

Conclusion

To conclude this work I look back at the questions I stated earlier:

• How can a web site in SharePoint be implemented to fit mobile devices by a person who knows nothing/little about SharePoint?

With some follow up questions:

• How do you implement a web site to fit multiple platforms?

• Is there a way to implement web sites in SharePoint without knowing that much about SharePoint?

When implementing web sites for multiple platforms one of the first things to look at is the content. Depending on what the content is there are different things to adjust. When you know what needs to be adjusted you need to find a solutions for the different platforms. In my case I found out that the navigation needed adjustment and then I looked at different solutions of how to implement it on a mobile device.

The technology used should be responsive design to reduce the amount of work when creating and maintaining web sites. It also offers the possibility to give a smooth transition between different device widths. It will work on all devices regardless of the platform as it adjusts depending on the width of the device.

The concept of mobile first should be used to make sure to focus on the content. This also makes sure the mobile will get its design and it will not be forgotten or postponed.
To get all this to work in SharePoint I use an approach where I let SharePoint convert my HTML, and then make all design with CSS and JavaScript. There will be some conflicts between the stylesheets and therefore I need to put some time into finding and fixing these conflicts.
Bibliography


På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Erik Claug