Phuc Doan

Practical Responsive Web Design

Helsinki Metropolia University of Applied Sciences

Bachelor’s Degree

Media Engineering

Practical Responsive Web Design

April 2017

Author(s) Title Number of Pages Date	Phuc Doan Practical responsive web design 37 pages April 2017
Degree	Bachelor of Engineering
Degree Programme	Media Engineering
Specialisation option	Mobile Application Development
Instructor(s)	Merja Bauters, Principal Lecturer

The purpose of this study is to provide feasible information to designers and developers to improve the process of building responsive websites. The study examines issues in web design and development projects and customers' needs based on the data gathered from the market of handheld devices to elaborate key principles and patterns of responsive web design. Sample codes and illustrations are included in the thesis to demonstrate ideas and practical techniques of each method that was used.

Furthermore, two case studies are described as illustrative examples to show how the discussed methods were applied in practice. The conclusion is that there are advancements in the process of building responsive web pages to meet customers' demands. In the end, there are always better approaches for improvement in Responsive Web Design as the supported technologies are evolving constantly.

Keywords	Responsive design, pattern, principles, mobile first, relative units

Contents

1 Introduction

Currently, in addition to desktop computers, there is an excessive number of handheld devices such as mobiles and tablets that are used to get access the Internet. Thus, websites need to be responsive so they are viewable regardless the viewing device. Although Responsive Web Design is widely known as a feasible solution, the process to design and develop a responsive web page has not been optimal. Designers and developers are having difficulties in converting static web pages to flexible ones.

Understanding this as a problem, this study is set to elaborate on key principles and patterns that are commonly used to build responsive web pages. This study will focus on aspects that are identified as common problems while commencing web design projects. Having insights on these principles and patterns will serve as a strong starting point for designers and developers to build responsive web pages better and faster to meet the demands of a customer.

To show the readers how the key principles and patterns in Responsive Web Design are applied practically, this study will include two case studies which describe the process to enhance the responsiveness in a realistic web design and development project.

2 Responsive Web Design

In this study, the goal is to show a reason why modern websites need to be responsive. An outline of the evolution of the characteristics of websites and the process of designing a website is given. The past is the basis for the present trend which is the responsive web design. Statistically, the usage of handheld devices and discussion of the advancement of mobile devices are main reasons for the essentiality of responsiveness in web design. Besides, Cascading Style Sheets (CSS) has been one of the critical forces constantly pushing the boundary of the World Wide Web. It has affected responsive web design directly. This section will include a short introduction to it.

Even though the term World Wide Web was invented for over 20 years ago, it is still considered as a new medium and the methodologies in web design keep changing and evolving continuously. Web design is a process of accumulating and arranging elements to form a unified digital space where end-users not only can interact, but also absorb the information [1].

Historically, web design started after NCSA Mosaic version 1.0 was publicly released in 1993 due to its capability of showing images and text in the same place [2]. According to David Siegel, the first-generation websites are described as a straightforward presentation containing images and text and the different segments of the page were likely to be displayed by bullet points and horizontal dividers [3]. However, due to the common 16 colours resolution used at the time, the colour range in one website was quite limited which lead to bleak appearance across the websites in this era. As a result, the first wave of websites presented information plainly without providing any specific interactions with the audience.

With the evolution of HTML through the extensions, which was implemented by Netscape in 1995, web designers and developers started to create innovative experiences for the audience via the web [3]. Since the 32 colours resolution monitor greyish backgrounds were replaced by colourful images, the content was organized in a more mindful way which improved and encouraged interactions between the users and the web itself. Unlike descriptive texts used in the first-generation to provide the information ordinarily, this time illustrative metaphors were adapted to create easy understanding, resulting in better user experience.

HTML was not alone able to enrich the experience on the web. It did not meet the demands from people on controlling how the web documents looked. In order to overcome the constraints of HTML, people wrote HTML in a non-semantic way which is not considered as a good practice. CSS came into the scene as the means for designers and developers to leverage the World Wide Web to a new ground. [4]

The first draft of the CSS was called Cascading HTML style sheets. It was published by Håkon W Lie in 1994. Internet Explorer was the first browser to implement CSS in August 1996. The core idea of CSS is to provide a web writer a tool to style the document. [4.] It allows people to change the size and the colour of the font or the background image of a specific element within the web document. This is illustrated in listing 1.

```
h1{
font-color:#010101;
font-size:12px;
}
div {
Background-image: url(‘sky.jpg’);
}
```

Listing 1. Setting font colour and size for h1 tags in HTML and changing the background of the tag.

CSS also has provided a mean to separate the stylistic information and the content of the web document, giving HTML back its semantic value, as it should be done [8]. As an example, when building a 2-column layout without CSS, the <table> tag in HTML has to be used to render the columns. This is a non-semantic way since the purpose of the tag is to build a data table. Fortunately, with CSS, users are handed various ways to achieve the goal without having to manipulate the original HTML structure. As listing 2 illustrates, the layout is constructed by building a table with one row and two columns. I

```
<table>
        <tr>
        <td>First Column </td>
```

```
          <td>Second Column </td>
          </tr>
<table>
```

Listing 2. A basic 2-column layout using only HTML

In contrast, in listing 3, the HTML is set up as elements that are in the semantical correct tags and the result of the layout is achieved in CSS separately by positioning them side by side by setting “float” property.

```
/* HTML Structure  */
<div id=”container”>
<div id=”column-1”> <p> First Column </p></div>
 <div id=”column-2”><p>Second Column </p></div>
</div>
/* CSS Style Set */
#column-1{
width:50%;
float:left;
}
#column-2{
width:50%;
float:left
}
```

Listing 3. A basic 2-column layout using HTML for structural mark-up and CSS for positioning web elements.

Basically, the process of building a website can be split into two phases: design and development. In the design phase, the visual style is defined using visual elements crafted by the designers. After that, development team will start to work to translate the materials from designers into HTML and CSS code and the result is a web document that users can access and interact through web browsers. Due to the fact that the majority of users in the 1990s were using desktop monitors to access the web, web pages were designed to be fitted only on one medium, with different screen sizes.

Recently, mobile phones, especially smartphones, have been a fast-growing market and from the business perspective, having websites that are compatible with browsers used on mobile devices brings numerous opportunities to expand the business and gain profit. The number of shipped laptops, notebooks and desktop computers has dropped below shipped smartphones since the last quarter of 2010. In other words, people prefer mobile devices to computer desktops when visiting web pages and using applications. For instance, in November 2010, it was stated that the amount of connections to web-based email sites had dropped by 6% and the connections to mobile email sites had increased by 36%. Furthermore, large digital companies such as Paypal, Ebay and Google have already reaped positive results from their operations through mobile channels. [5.]

Initially, the trend began in 2006 when Apple introduced the first iPhone. It was the first mobile device that allowed users to browse the web comfortably. Thus, according to AT&T, the mobile traffic data went up by 4932% between 2006 and 2009. Additionally, the faster the mobile network is, the bigger the amount of mobile data traffic is. Besides, the price of smartphones devices and mobile data plan are decreasing which allows people who cannot afford desktops or laptop computers to view the web via mobile devices. Evidently, the sales of smartphones grew by 96% in 2010 compared to the previous year. [5.]

The mobile devices are not only cheaper, but they also evolve rapidly. This has opened multiple gates for designers and developers for building different kinds of services for the users of mobile devices. For instance, smartphones support multi-touch gesture technology. Multi-touch means that users use multiple fingers to perform different actions on the devices. New patterns in interactions besides single-touch action has been invented so that the experience of using mobile applications has become more intuitive. For example, the “zoom” action can now be performed by moving two fingers simultaneously in opposite direction, and the action “zoom in” and “zoom out” are defined by the relative distance between the two fingers. As a result, instead of using common peripheral devices (mouse and keyboards) to tap the control elements on a graphical user-interface, users now perform zoom action as a natural interaction that is suitable for mobile experience. A mobile device’s location is detected natively and designers and developers can access information from a browser’s API to provide relevant information based on the user’s location. [5.] In fact, location detection has been implemented to smart search features on the web so a website will filter the result related to the user’s current location.

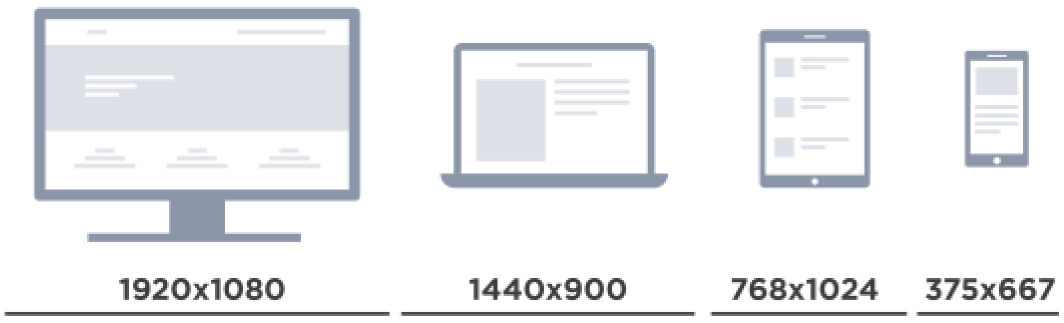


Figure 1. Collection of devices and screen dimensions. Data gathered from Responsive Web Design [6,144].

Figure 1 presents different screen sizes that are widely available in the market. The current set of handheld devices also has a wide range of screen dimensions. Thus, building web pages the contents of which will be accessible and consistent across different devices is critical for success. Businesses that rely on web documents that are designed and developed to be static will properly be crippled as customers are not able to view their website properly on their device. To deal with the uncertainties among the hardware landscape and to deliver a coherent experience to the users, responsive web design is the solution. [6.]

Responsive Web Design is a mastery to the make one website adapt to all screen dimension of devices [6]. Before Responsive Web Design was considered as a trendy contemporary solution in web development, there were two common solutions for the differences of the devices. The first solution was to build different versions of one web page for each specific device. For example, besides creating the version of a desktop computer, developers also built separate web pages that were only available for mobile and tablet viewers. This solution relied heavily on a technique named User Agent Sniffing. The users are provided the webpages based on the device they are using. The server knows which version should be delivered based on the information retrieved from the users. The retrieved information reveals what kind of device and which browser are in use in the client’s side. This method proved to be unreliable in some cases because user agent information can/could be modified by the user’s browsers and networks.

[7,48.] The second solution is to build a native application to represent the same information appropriately on each device. The advantage of native applications is their capabilities to utilize hardware services which mobile web browsers usually do not have access to. However, both solutions have identical disadvantages in increasing the cost of production for building different versions and using different programming languages. [1.] As the client-side technology like HTML, CSS and Javascript have been evolving, it grants developers and designers better tools to access the native services and to detect the dynamic environment of the web browsers. As a result, responsive web design has become more approachable.

Even though people understand the benefits and motivations of Responsive Web Design for modern web development, to enhance the flexibility of the web documents, designers and developers should be prepared and update their skillsets since the landscape of the World Wide Web changes constantly. In the next section, I identify and discuss critical principles and patterns in Responsive Web Design that are utilized in web development in general.

3 Principles and Patterns

Designing and developing web documents responsively is a holistic solution which allows flexibility of the web pages. Furthermore, Responsive Web Design opens opportunities for designers and developers to revise their workflow for delivering better user experiences. This section will analyse key principles and patterns which are used commonly in Responsive Web Design.

3.1 Mobile First

The mobile market is exploding, meaning people have many options to choose from and new devices are released regularly with better hardware and cheaper price. It is reported that there were approximately 7.9 billion mobile devices in the world in 2015. The majority of devices are smartphones. The increase of smartphones was half a billion from 2014 to 2015. Hence, Luke Wroblewski put forward a new design paradigm called “mobile first”. It is suggested that mobile experiences should not be considered as an alternate solution in a web design project but rather be designed and developed as a primary focus from the beginning of a web design project. The purpose of this paradigm is to take advantage of the excessive growth in the market of handheld devices. [5.]

Since mobile devices have their own characteristics compared to desktop computers, designing web pages for mobile experiences requires the developer and the designer to re-evaluate the approaches in building the web pages. Firstly, one of the most noticeable distinctions between desktop computers and mobile devices is screen size. The space for contents in mobile devices has been cut down approximately 80% compared to desktop computers as the common dimension of a desktop is 1024x758 pixels and the resolution of popular smartphones is 320x480 pixels. Due to the limitation in space, the functionalities of web pages should be selective. Designers and developers need to know what the customers’ and businesses’ needs are for the website. This is the key solution to determine what the critical features are that should be available and what should be prioritized in the web page as the mobile environment does not have enough rooms for irrelevant details. Therefore, the core services of the web pages are focused and defined unambiguously from the beginning and when scaling the web pages to other devices, the core services remain the same which allows customers to have equal experiences no matter what devices they are using to view the web pages. [5.]

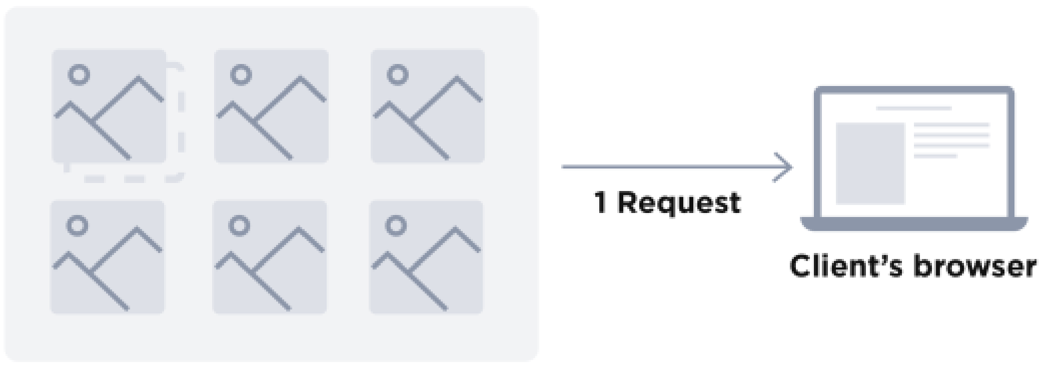


Figure 2. Bundling multiple graphical assets into one file to reduce the amount of request sending to the client's browsers. Data gathered from Mobile First [5,73].

Secondly, even though mobile networks can be accessed from nearly anywhere, their speed is not considered to be as fast as the Internet connections people use on desktop computers. The mobile network is more expensive than the desktop network. Consider this constraint, the experiences provided to customers using mobile devices have to be optimized. Specific methods should be taken into account during the development and design process in order to increase the performance of the web pages. As figure 2 shows, a number of HTTP requests sent to customers' mobile devices are reduced since multiple images are bundled into one image. As listing 4 illustrates, the usage of CSS3 properties is provided in case the web pages are accessed from old browsers. CSS3 is available natively within modern browsers to reduce the needs of using images, which serve the same purposes and which, in some cases, are a fall back. [5.]

```
/* Adding rounded corners */
#element-1{
border - radius: 4px 4px 4px 4px;
}
/*Adding gradient background */
#element-2 {
background: #ED4264; /* fallback for old browsers */
background: linear-gradient(to left, #ED4264 , #FFEDBC)
}
/*Adding box shadow */
#element-3{
box-shadow:  0 4px 8px 0 rgba(0,0,0,0.2)
}
```

Listing 4. Using CSS3 properties to add rounded corners, gradient background and box shadows to HTML elements.

The benefit of optimizing a web page’s performance on mobile devices upfront is that developers and designers do not have to worry about performance issues when translating the design beyond mobile devices.

The interactions are considered as a bridge when connecting the users to the product. Crafting proper interactions on the web page helps delivering successful experiences. As it is stated above, the nature of mobile devices is different from desktop computers so usage patterns that are employed to design interactions for desktop computers should not be applied directly to mobile devices. Foremost, the mouse and keyboard are the primary means for users to interact with the web; however, as devices become smaller, “touch” becomes the way to input actions. Even though track pads and keyboards are available on some mobile devices, the number of users preferring touch-based interactions has been increasing gradually, as figure 3 shows. [5, 68]

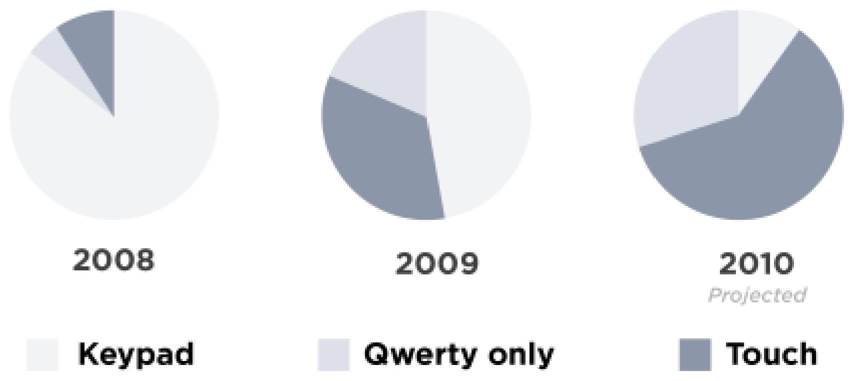


Figure 3. Transition to touch-based devices on Nokia smartphones. Reprinted from Mobile First (2011) [5,68].

Naturally, the human fingers are less precise compared to a pixel-based pointer controlled through the mouse [1]. As the screen sizes are smaller on the mobile devices, the area of the touch target is larger in order to reduce inaccuracies within the interaction between users and web documents. Therefore, the usability is assured when the minimum height of the touch target is 48 pixels in a 72 ppi screen as it is illustrated in figure 4 [9]. On the other hand, the touch target area should always be larger than the original area of the button.

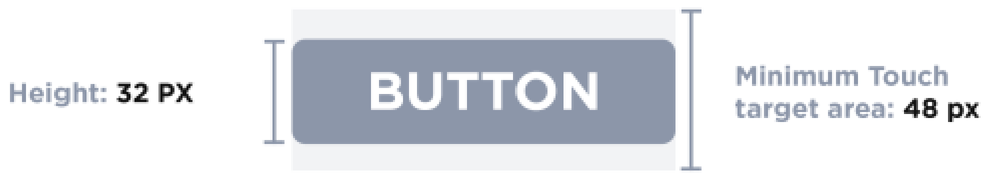


Figure 4. Metrics for designing a button on mobile devices. Data gathered from Material Design [8].

Since the ergonomics of webpage system has changed from desktop to mobile, organizing web elements on the screen has to be re-evaluated to maximize usability for all users across multiple devices. Statistically, 10-30% of people are left-handed and it is observed regularly that mobile devices are used in one hand and that the right thumb finger is used to interact with the device [5]. Therefore, a critical call-to-action should be placed either in the bottom or centre of the screen as it is showed in figure 5 so users would take less efforts to complete the action.

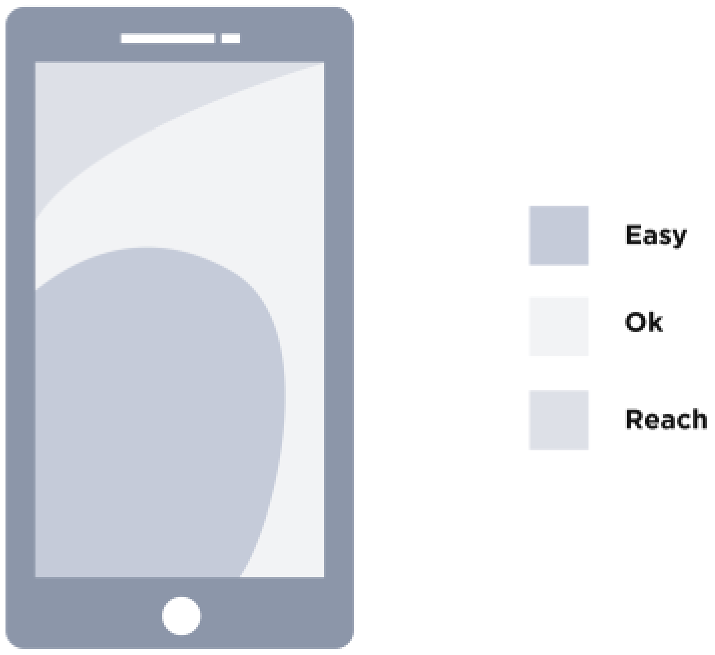


Figure 5. Comfortable areas while using smartphone. Modified from Mobile First (2011) [5,73].

“Hover” is a common event on desktop interface when the user’s cursor moves over a specific element on the web. Following that, an action will be triggered to provide feedback to users. For example, as it is illustrated in Figure 6 additional information will be revealed when users hover on top of an image on the web. However, on handheld devices, especially touch-based devices like iPhone and iPad, the hover action is obsolete since there is no cursor to use. Therefore, the purpose behind all hover actions that are used on the desktop has to be redefined when designing for mobile devices. There are three common alternative solutions to this problem and the they are applied based on the value of the content in a specific context. Firstly, if the triggered event shows primary actions or relevant information to users, these actions and information should be place directly on the page. Secondly, if the hovered information is considered as an expansion, a separate screen should be created. Lastly, if the value of the hover effect is inconceivable, it is strongly recommended to let it go in order to reduce the complexity the page’s user interface. [5.]

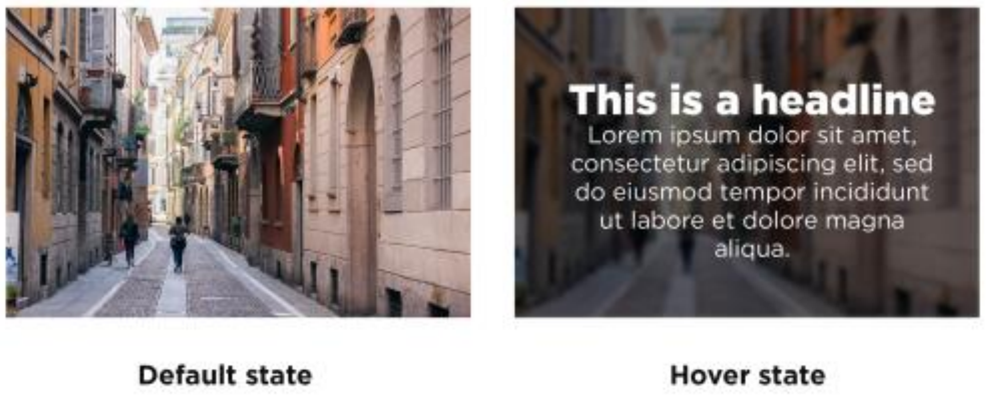

This is a headline
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Default state
Hover state

Figure 6. Default and hover state of an element. Data gathered from Mobile First [5,79].

3.2 Relative Unit and Media Query

CSS has been used to set the styles for all the elements on the web, and new features have been still implemented to enhance the responsiveness to the web pages. Therefore, this section will discuss in detail the usage of the common CSS features which are practical in Responsive Web Design.

Instead of using fixed typed values for measurements in CSS, relative units are considered more relevant since all elements on the web pages are usually required to be scalable.

Table 1. Common absolute units and relative units in CSS. Data gathered from Mozilla Developer Network [9].

Name	**Type**	**Description**
Pixel (px)	Absolute unit	1 pixel = 1 point
Point (pt)	Absolute unit	1 point = 1/72 inch
Inch (in)	Absolute unit	1 in = 2.54 centimetres
Em	Relative unit	Relative to the font-size of the element
rem	Relative unit	Relative to the font-size of the root element which is the <html> in this case

As it is illustrated in table 1, Pixel (px) is one of the most popular measurement units that is used in CSS. However, the webpages nowadays are viewed on various types of devices so building web components that rely on static units as pixel is not considered practical. Thus, it is recommended to translate absolute values to relative values in order to maintain the proper proportion of the web element when the webpages are viewed on multiple devices. For instance, in the example listing 5 gives, the goal is to create a two-columns layout, so instead of defining the width of each column in pixels, it is practical to use percentage (%) for preserving the proper proportion between the two components regardless the devices they are being viewed on.

```
/* Setting absolute value */
div#sidebar {
width: 320px;
}
div#content{
          width: 768px;
}
/* Using relative values */
div#sidebar {
          width:30%;
}
div#content{
          width:70%;
}
```

Listing 5. Using relative unit to maintain the proper proportion of the components.

In addition to the percentage, em and rem units are used to maintain the proper proportion between the web elements. Em and rem are used for the web browsers to determine the actual measurement of the web component in pixel. The rem unit is calculated based on the default font-size of the root element which is an/the <html> tag and the default font-size value is 16 pixels. For instance, in order to express pixel unit in rem unit, the formula below is applied:

Target / 16 = Result (1)

“Target” is the value in pixels and “Result” is the value in rem. As 24 / 16 = 1.5, the value that should be used in CSS is “1.5 rem”. The relative units em are calculated with this formula:

Target / Context = Result (2)

The meaning of the parameters “target” and “result” in (2) are the same as they are in (1). Instead of dividing the pixel-based value by 16, a new parameter “context” is defined by the font-size of the current element in the HTML structure. [4,21.] For instance, in listing 6, since the font-size values of container-1 and container-2 are different so the value of the margin also expresses dissimilarity in em unit.

```
#container-1{
          font-size: 16px;
          margin: 1.5 em; /* 24 / 16 = 1.5 */
}
#container-2{
          font-size: 18px;
margin: 1.33333333 em; /* 24 / 18 = 1.33333333 */
}
```

Listing 6. Relative value in ems are varied depending on the font-size property of the container.

Responsiveness in the web pages is leveraged by applying a media query in CSS as an anticipation to the variety of the devices that are used by the users. Responsiveness was introduced as media types which were introduced in CSS2 specification. Its goal is to differentiate stylesheets based on different media and device types. For instance, in listing 7, the generic.css is always loaded regardless the device or media it is on because the media attribute is set to “all” which is defined as a supergroup to all the parameters. In the following rows, there are three stylesheets and each one is assigned to a specific media or device so when the page is accessed from a screen-based device, the page only loads the file screen.css and ignores the rest.

```
<link rel=”stylesheet” href=”generic.css” media=”all”
/>
```

```
<link rel=”stylesheet” href=”screen.css”
media=”screen” />
<link rel=”stylesheet” href=”mobile.css”
media=”handheld” />
<link rel=”stylesheet” href=”print.css” media=”print”
/>
```

Listing 7. Using different stylesheets based on media type.

Alternatively, multiple stylesheets can also be incorporated into one CSS file to reduce the number of requests sent to the server. It is done by using @media statement in CSS as it is demonstrated in listing 8.

```
@media all {
          /* Define your stylesheet here*/
}
@media screen{
          /* Define your stylesheet here*/
}
@media print {
/* Define your stylesheet here*/
}
```

Listing 8. Using @media to implement multiple stylesheets into one CSS file.

However, due to the advancement of mobile devices, the “handheld” value in the media attribute cannot detect all the browsers on the modern mobile devices and almost all mobile web browsers are treated as screen-based media. Thus, using only media types in media query could not fulfil the goal of enhancing responsiveness to the web pages. [6,73-74.] Fortunately, media queries were introduced as an extension to media type in the CSS3 specification to solve the problem. In addition to identifying what kind of media, relevant properties of the web browsers and devices can be also examined by using the media query in CSS.

```
@media screen and (min-width: 1024px) {
        /* Define your stylesheet here*/
}
```

Listing 9. An example of using media query in CSS

As it is illustrated in listing 9, the structure of a media query statement includes two parts: @media screen which is already explained above, and (min-width:1024px) which is defined to inspect the width of the browser that the web is rendered on. In short, the stylesheet is applied when the media is screen-based and the width of the browser is at least 1024 pixels. If the conditions are not fulfilled in the media query statement, the stylesheet within the query is simply ignored by the browser. Many features of the device can also be queried and they are described in table 2. As a result, a collective of a/the target device's characteristics can be defined within one query which is considered as a feasible feature to enrich the responsiveness on the web pages [6].

Table 2. A list of device features that are compatible with media queries. Modified from Responsive Web Design (2011) [6,76].

Feature name	**Descriptions**
width	The width of the display area
height	The height of the display area
device-width	The width of the device's rendering surface
device-height	The height of the device's rendering surface
orientation	Acceptance of (?) portrait or landscape value
aspect-ratio	Ratio of the display area's width over its height
device-aspect-ratio	Ratio of the device's rendering surface width over its height

Using multiple media queries allows developers and designers to define different breakpoints, which ensures the integrities of the content when it is viewed on different devices. Additionally, the media query and the “mobile first” paradigm can be combined in order to increase the responsiveness for web pages. As it is demonstrated in listing 10, the stylesheets for the mobile scenarios are built as default in the CSS. As the resolution is increasing, media queries are used to define the characteristics of the layout at different breakpoints. Thus, users who are using old browsers that do not support media queries can also view the web pages.

```
/*Default layout */
.section {
          margin: 0 auto;
          max-width: 700px;
}
/* Medium screen layout */
@media screen and (max-width:991px){
          /*Custom stylesheets*/
}
/* Large screen layout */
@media screen and (max-width: 1199px){
          /*Custom stylesheets*/
}
/* Extra large screen layout */
@media screen and (min-width: 1200px){
          /*Custom stylesheets*/
}
```

Listing 10. Apply “mobile first” paradigm with media query

3.3 Flexible images and videos

In addition to text, images and video elements are also served as essential means to convey a message from the pages to the users. As the web pages go responsive, videos and images have to be transformed from static to fluid in order to be compatible with the responsiveness of the pages.

One of the most common problems when dealing with images is that the size of an image is not match with the size of the container which leads to a situation where the rendered image on the page is bigger or smaller than it is intended to be. Thus, setting max-width to 100% for the image, as it is demonstrated in listing 11, is one of the fundamental things that should be done to enhance the fluidity into the images on the web. Setting the max-width to 100% to images will prevent them from surpassing the containing element since the image's width is adapted according to the width of the container and since the container is fluid with relative units, whenever the browser resizes itself, the ratio of the image is kept intact. [10,57-58.] Moreover, the maximum width is applicable with rich media and video.

```
<div class="container">
    <img src="img/dog.jpg" alt="A Dog">
</div>

.container{
    width: 42%;
}
img{
    max-width:100%;
}
```

Listing 11. Using maximum width to create fluid images and videos on the web.

In addition to separate images, new properties are provided in the CSS to create flexible background images on the web documents. The background images are displayed natively in their resolution and the dimension is constrained by the width and the height of the containing element. Moreover, default characteristics of the background images can be modified through background-size property. As it is demonstrated in listing 12, the fixed and auto values are combined to make sure the image scales proportionally. Alternatively, the relative unit such as percentages are assigned for scaling relatively to the containing element. Furthermore, background images are displayed fully and proportionally in the containing element by using the value cover for the property background-image. In this case, the image background is scaled based on the smaller value between its width and height and the overflow of the image will be hidden. [10,67-70.]

```
.container {
        background-image: url('example.jpg');
        background-size: 600px auto;
    }
    .container {
        background-image: url('example.jpg');
        background-size: 50% 50%;
    }
    .container {
        background-image: url('example.jpg');
        background-size: cover;
    }
```

Listing 12. Utilizing background-size values to enhance flexibility to the background images.

3.4 Navigation

A navigation bar is a critical segment in every web page as users rely on it to traverse within the site hierarchy. Thus, building a flexible navigation bar that is functional in multiple devices is considered as an important step toward a complete responsive website. The goal of this section is to explain patterns that are deemed as solutions to the challenges in designing a navigation bar responsively for the web documents.

A navigation bar is a complex network of options and sometimes it has multi-tiered selection, which brings a vast amount of information to web pages. While designing for mobile experience, adding the content for a navigation bar for mobile devices is the common issues due to the disproportionateness between the amount of content and small screen size of mobile devices. As a result, the viable solution is to hide all the information from the navigation and reveal it only when users trigger an action indicating they need to use the navigation bar. In this method, in addition to all the list elements of the navigation bar, a new element is added as a trigger to display the navigation on small devices. A Javascript function is assigned to it so whenever users trigger the event, the function will add necessary information to the HTML elements as it is shown in listing 13. [10,18-19.]

```
<ul class="navbar" id="myNavbar">
                    <li class="icon">
```

```
                        <a href="#"
onclick="goResponsive()">&#9776;</a>
                </li>
                <li><a href="#home">Home</a></li>
                <li><a href="#news">News</a></li>
                <li><a
href="#contact">Contact</a></li>
                <li><a href="#about">About</a></li>
  </ul>
<script>
 function goResponsive() {
                var x =
document.getElementById("myNavbar");
                if (x.className === "navbar") {
                    x.className += " responsive";
                } else {
                    x.className = "navbar";
                }
            }
</script>
```

Listing 13. Adding an element to trigger the menu and a Javascript function to append necessary information for CSS.

In the CSS, the triggering element is hidden by default and only visible on a smaller device. It is achieved by using a media query / media queries as it is demonstrated in listing 14. Furthermore, inside the media query, new rules are defined to hide all the menu elements and display them stacked up vertically when users evoke the trigger.

```
ul.topnav li.icon {
        display: none;
}
@media screen and (max-width:680px) {
        ul.topnav li {
            display: none;
        }
        ul.topnav li.icon {
            display: inline-block;
        }
        ul.topnav.responsive {
            position: relative;
}
 ul.topnav.responsive li {
            float: none;
            display: block;
        }
        ul.topnav.responsive li a {
            display: block;
            text-align: left;
        }
 }
```

Listing 14. Using a media query to toggle the visibility and orientation of the elements in the navigation bar.

Furthermore, instead of dropping the menu elements from the navigation bar as it is demonstrated by listings 13 and 14, the menu can be slid from the left to the screen, which is defined as an off-canvas navigation bar. For example, as figure 7 shows, as the user clicks the trigger button on the left, a menu slides from the left presenting all the possible destinations for the user to navigate. [10,28-29.]

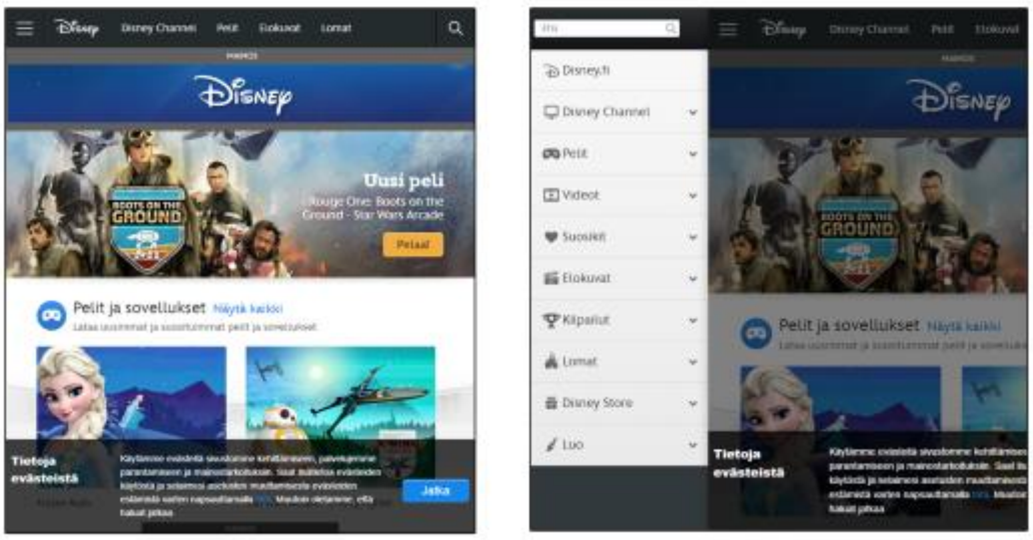


Figure 7. Off-canvas menu on Disney.com. Data gathered from Disney’s homepage [11].

3.5 Tools

Understanding basic principles and patterns enable developers and designers to build responsive web pages properly. However, the market is more demanding and the budget to build responsive web pages is disproportionate to the time frame of the projects. Thus, third-party frameworks and tools are utilized. This section is meant to show an overview of the common frameworks and tools that are used to create responsive web pages and to discuss the advantages in using them.

Many front-end frameworks have been developed recently to create a responsive web without building it from scratch. Using a framework to build responsive web pages provides many advantages. As it is already stated, the groundwork for responsiveness is bundled within the framework so developers do not have to reinvent the wheel which is considered time-efficient. In team environment, as the web pages are built upon the same codebase, different members can join and continue the project without spending a great deal of effort to adapt with protocols of the project. Furthermore, since the frameworks are built and maintained by a team of professional developers and designers, the fundamental functionalities are optimized and developed using modern practices. Besides, bugs and errors are reported by users globally so frameworks are updated regularly.

Bootstrap is one of the most popular frameworks for responsive design. It was produced by employers at Twitter as a style guide tool for company internal application and publicly released in August, 2011. Currently, the third version has been released the codebase of which is rewritten to improve performance and compatibility. It also employs the mobile-first approach to enhance responsiveness into the web pages. The fourth version is now in alpha testing and is predicted to be fully released in 2017. [12.]

In version 3 of Bootstrap there are four major predefined breakpoints to represent four tier devices that are available on the market: phones, tablets, desktops and widescreen desktops. Since Bootstrap is employed the “mobile-first” paradigm, stylesheet for mobile devices is defined as a default as it is shown in listing 15.

```
/* Extra small devices (phones, less than 768px) */
/* No media query since this is the default in Bootstrap
*/
/* Small devices (tablets, 768px and up) */
@media (min-width: 768px) {...}
/* Medium devices (desktops, 992px and up) */
@media (min-width:992px) {...}
/* Large devices (large desktops, 1200px and up) */
@media (min-width: 1200px) {...}
```

Listing 15. Predefined media queries in Bootstrap 3.

Within Bootstrap, a grid system is established for users to build scalable responsive layouts. In order to get the elements into the grids, all elements have to be wrapped inside the containing element defined by Bootstrap. In detail, there are two types of containers in Bootstrap: “.container” which is used for the fixed-width element and “.container-fluid” which is for full-width element. The grid system is divided into 12 columns and the allocated horizontal space for an element on the grid is defined by using predefined classes provided in Bootstrap. For instance, listing 16 demonstrates how to create a simple layout structure consisting of three sections by applying Bootstrap’s grid system. In the first section, there is only one element that is spanned across 12 columns. Then, in the next section, there are three elements that would need to line up horizontally and equally. To do that, they are set to be allocated in four columns for each. Lastly, the final sections consist of two elements and the first element is moved to the left by two columns by using the prefix “-offset”. [13.]

```
<div class=”container>
<div class=”row” id=”first-section”>
<div class=”col-md-12”>Lorem ipsum</div>
</div>
<div class=”row” id=”second-section”>
<div class=”col-md-4”> One </div>
<div class=”col-md-4”> Two </div>
<div class=”col-md-4”> Three </div>
</div>
<div class=”row” id=”third-section”>
<div class=”col-md-4 col-md-offset-2”> One </div>
```

```
<div class=”col-md-4”> Two </div>
</div>
</div>
```

Listing 16. Using Bootstrap’s grid system to build a simple layout with three different structures.

The grid behaviour is adjusted responsively based on the key breakpoints defined in listing 16. In Bootstrap, since there are four major breakpoints, there are four types of grid classes respectively. To allow the elements to be displayed differently on multiple devices, predefined classes can be combined as is demonstrated in listing 17. For the need to show and hide elements between different breakpoints, utilities classes “. hidden-*-*” and “. visible-*-*” are available to accomplish the given task.

```
<div class=”col-md-4 col-xs-12”> …. </div>
```

Listing 17. Display an element differently on multiple devices.

Bootstrap 3 supports the/a responsive approach for the navigation bar and image. To enhance the flexibility of images, Bootstrap provides a predefined class named .image-responsive which allows the element to be scalable according to the parent elements. As for the navigation bar, it is collapsible when the viewport gets smaller and will line up horizontally when the screen is viewed on a larger display. The example of how images and navigation are declared in HTML using Bootstrap’s component with the predefined classes is demonstrated in listing 18.

```
<nav class=”navbar navbar-default”>
<div class=”container”>
          <div class=”navbar-header”>
<!-- Navigation bar header where the toggling button
is defined -->
          </div>
          <!-- Define the toggling list when it is
displayed on viewport smaller than 768px-->
<ul class=”nav navbar-nav navbar-right’>
          <li> <a href=”#”>Link </a> </li>
<li> <a href=”#”>Link </a> </li>
<li> <a href=”#”>Link </a> </li>
          </ul>
```

```
</div>
</nav>

<img class=”image-responsive”> </img>
```

Listing 18. Responsive image and navigation bar using Bootstrap 3.

4 Case Study

In this section, case studies will be presented as a mean to illustrate how the principles and patterns discussed in the previous chapter are applied to realistic projects to meet the demands of customers. All the case studies were carried out in a marketing agency which is in Helsinki, Finland, and the author participated in the projects as a graphic designer and front-end developer.

4.1 Finnair

The subject in the first case study is a web page for Finnair, a Finnish airline, and the purpose of the page is to provide extensive details of potential destinations around the world to encourage customers to purchase services from the airline. Since there is excessive number of people using handheld devices to get access to the internet, the page has to be developed and designed to be responsive. For the design process in this project, the author works/worked? under the counselling of the creative director in visual design and art direction at the marketing agency. Then in the development process, the author collaborated with a web developer.

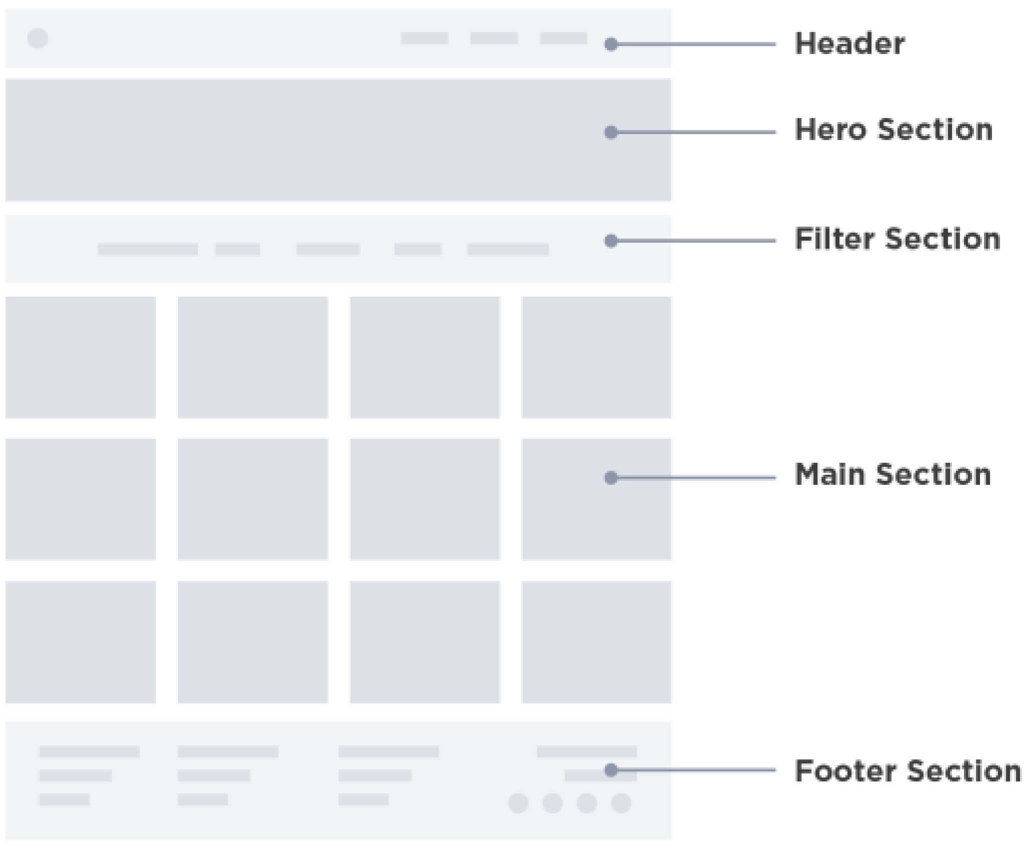


Figure 8. Overall structure of the page.

Before the process of visual design and front-end development started, the structure of the page had been analysed and divided into five sections, as shown in figure 8. In this case, the header and footer were used the universal components which were provided by the client. The implementation of the rest of the page was carried out by the marketing agency. As responsiveness was one of the vital requirements for the project, the design and development team decided to approach this project using the mobile-first paradigm to maximize the responsiveness. Every key aspect of the page was designed for the mobile environment. As for the development side, inside the stylesheet file, styling information for devices that were larger than the mobile was defined within specific breakpoint. Moreover, mobile’s styling information was defined outside any specific breakpoint to ensure the page was always displayed properly even when users used browsers that did not support media-query. In this case, due to the time constraint in the project, only one breakpoint was defined for the desktop to show the full version of the page on a large monitor, and the basic structure of the CSS file is demonstrated in listing 19.

```
/*Styling for mobile devices  */
.section{
….
}
/*Styling for desktops */
@media (min-width:992px) { ... }
```

Listing 19. The basic structure of CSS file for the project.

In the hero section, the font size of the headlines and the text below were set using the rem unit in default mode to assure they are always scaled proportionally when the website is viewed on different devices. Besides, the minimum text font size is decided when designing the mobile screen first to make sure the legibility of the content on all devices. Moreover, the background image ratio is maintained by setting the background-size. Thus, in figure 9, the hero section is scaled proportionally from mobile to desktop.

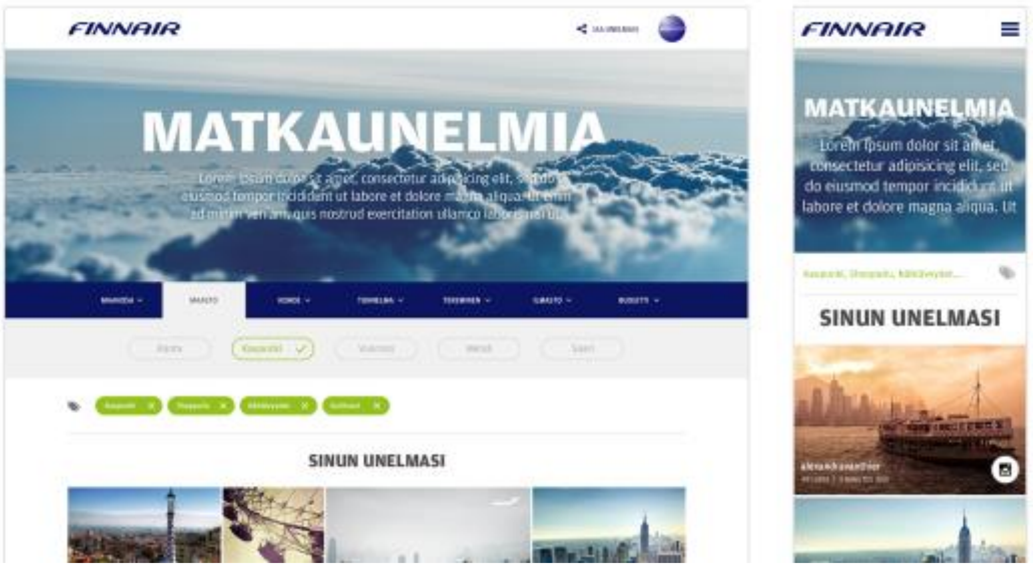

FINNAIR
MATKAUNELMIA
SINUN UNELMASI
FINNAIR
MATKAUNELMIA
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut
SINUN UNELMASI

Figure 9. The hero section on mobile devices and desktop computers.

The following section is created as an interface for users to choose their preferences. Foremost, to ensure the target area has enough space for users to perform interaction properly on mobile conditions, the area around the component is set to be at least 48 pixels as it is showed in figure 10. Furthermore, due to the limit space of mobile devices, the decision is made to hide all the options and only reveal them whenever a user users click the trigger as it is demonstrated in figure 10. From the development side, developers use the same techniques that are explained in section 3.4 which binds a Javascript functions to the trigger element. Whenever the function is triggered, it will change the display value of the menu to show all the elements. As the screen size becomes wider, there is more room to display the whole component fully.

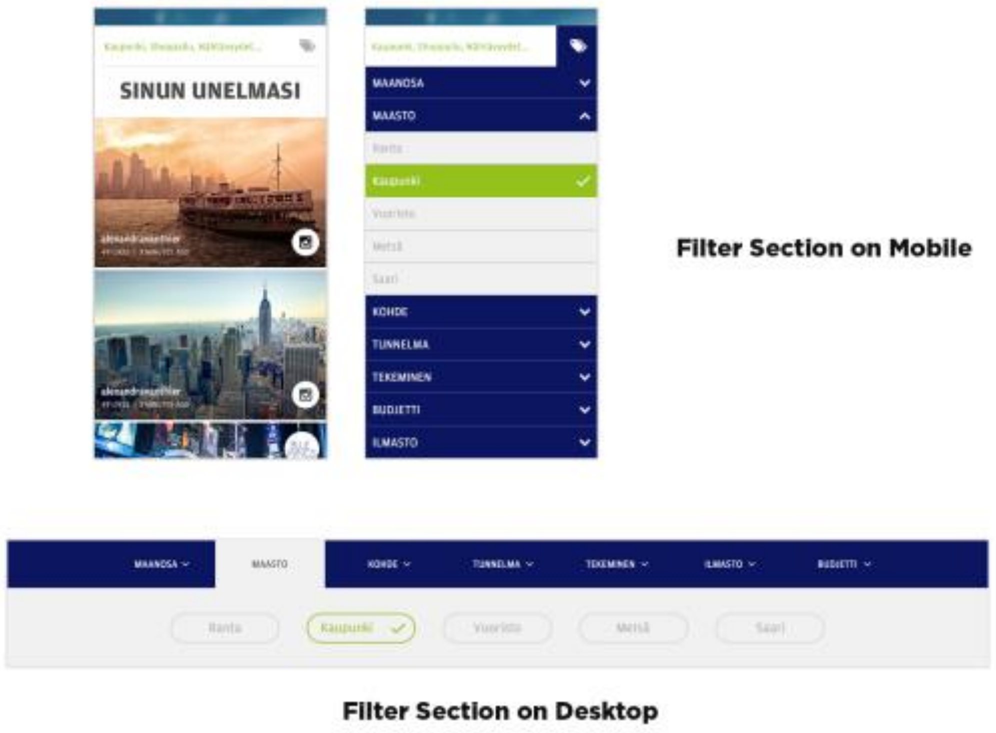


Figure 10. Enhancing responsiveness to the filter section of the page.

The main section of the page consists of a collection of images that are pulled from the filtering component based on the preferences provided by the user. Starting from mobile devices, the section is designed to have one image on each row to increase the area of the container. Thus, internal parts such as icons, headline and sub-headline have more spaces to be displayed on top of the image. Additionally, as mobile users regularly have limited amount of data in their network, the number of images that should be displayed initially is five. If users want to see more, they can request more images by pressing the button at the bottom. The images are scaled properly as their maximum width in the CSS file is set in percentages to accommodate with the container's width and height. Initially, since only one image is showed on each row for mobile experience, the max-width property is set to 100% to allocate the full width of the viewport. As the page is viewed on larger displays and there are more rooms horizontally, it is more feasible to display multiple images on a single row. In more detail, to enhance the diversity to/of (?) the image grid on desktop computers (?), it was decided that the number of images per row is ranged from two to four and their width can be varied. For instance, in figure 11, the first row contains four images and the value of the max-width for each item would be 25%, 10%, 40% and 25% respectively. On the second row, there are two images and one is a panorama so it is best to make it appear wider so the ratio is 70% and 30% for the max-width of each item.

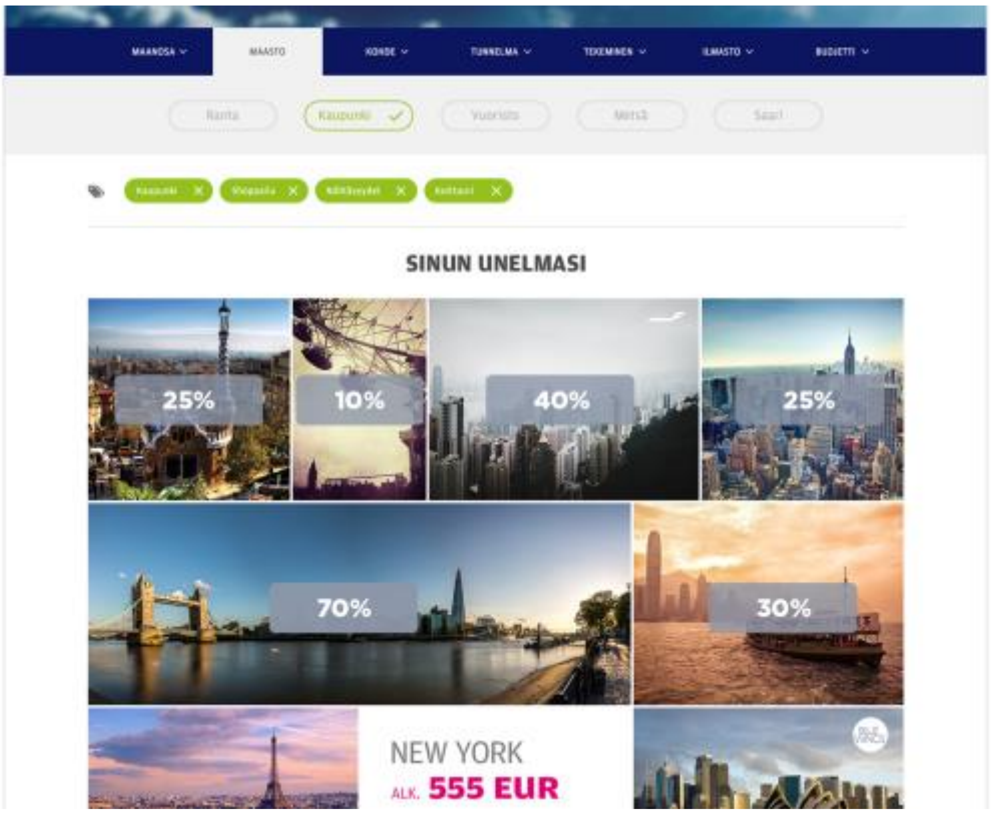


Figure 11. Unequal max-width between multiple items on one row for desktop computers.

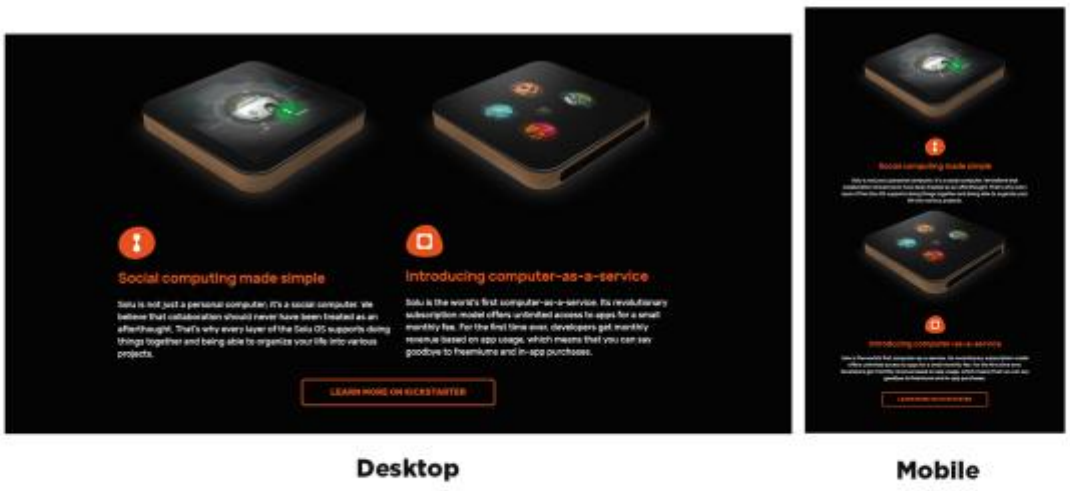
On mobile devices, the hover effect is removed completely as all the vital information of each item is displayed on top of the image by default. As the design goes to larger displays, the hover effect is defined inside the media-query so the information is only visible when users place the cursor on top of the image. The modal box is opened whenever users click or touch on the image and there are two possible layouts for it. First, one is a blog post which includes a headline and an article which have text and images in between. The second one is mainly about an image and a short description below. The typography within each layout is defined by using a relative unit to ensure it is scaled proportionally when being viewed on multiple devices.

At the end of the project, a responsive web page is produced to be shown as a demonstration to Finnair. The page flexibility is demonstrated successfully when switching from mobile to desktop devices. Applying the mobile-first paradigm reduces the workload when designing and developing web pages to work on multiple devices since all the problems are solved and key decisions related to the usability of the interface on the page are done from the first phase when using this kind of approach. On the other hand, the page is also viewable even on browsers that do not support media queries. However, due to the limitation of time and budget within the project, there are areas within the page that need improvements as well. For example, in addition to the breakpoint that is defined in listing 19, another breakpoint for a larger desktop screen and another one for tablet devices should be designed and implemented to increase the responsiveness of the page in overall.

4.2 Solu Machines

In the second case study, the subject was a landing page for Solu Machines, a startup company located in Helsinki, Finland. The purpose of the landing page is to provide information for the upcoming hardware from the company. Since the product is marketed globally, there is an excessive number of worldwide visitors meaning the page must be responsive in order to be viewed on multiple devices. During the project, the author served as a visual designer and front-developer under the directions of the creative director in the marketing agency.

The allocated budget of the project only allowed three days for the design and development phase and it was time-consuming to code a stylesheet file from scratch to implement the responsiveness to the web page. Thus, after analysing the overall structure and specific elements on the landing page after the design phase, Twitter Bootstrap version 3 was chosen to be the framework for the front-end development since it already has built-in functionalities that can produce a responsive web page within a short time.



Social computing made simple

Solu is not just a personal computer, it's a social computer. We believe that collaboration should never have been treated as an afterthought. That's why every layer of the Solu OS supports doing things together and being able to organize your life into various projects.

Introducing computer-as-a-service

Solu is the world's first computer-as-a-service. Its revolutionary subscription model offers unlimited access to apps for a small monthly fee. For the first time ever, developers get monthly revenue based on app usage, which means that you can say goodbye to freemiums and in-app purchases.

Figure 12. Two features displayed side by side on desktops and stacked on top one another on mobile devices.

First, the grid system from Bootstrap enables the elements to be positioned properly according to the visual design documents. For instance, in figure 12, there are two features of the product that need to be displayed side by side on desktop computers and when it comes to mobile devices these sections must be collapsed. As a result, listing 20 demonstrates how to accomplish the task by defining a 6-column section for each feature for large and medium displays and 12 columns for mobile devices. Inside each section, image and text elements are defined through multiple rows. Moreover, by adding the predefined class “image-responsive” to all images on the page, the images are scalable proportionally when being viewed on multiple devices.

```
<div id="subContainer" class="container">
            <div class="row">
                <div class="col-md-6 col-xs-12">
                    <div class="row">
                        <img        src="css/svg/solu-
social.png" class="img-responsive" alt="">
                    </div>
                    <div class="row">
```

```
                        <div class="col-md-12">
                            <img
src="css/svg/ic_social.svg"   class="   img-responsive"
style="width:77px;">
                        </div>
                    </div>
                </div>
                <div class="col-md-6 col-xs-12">
                    <div class="row">
                        <img        src="css/svg/solu-
collab.png" class="img-responsive" alt="">
                    </div>
                    <div class="row">
                        <div class="col-md-12">
                            <img
src="css/svg/ic_collab.svg"   class="   img-responsive"
style="width:77px;">
                        </div>
                    </div>
                </div>
  </div>
```

Listing 20. Defining a 6-column section for each feature.

In the sponsor section, all the sponsor's logos are bundled into one single bitmap file so when having multiple requests for the logos, the browsers only need one request to get all the logos, which improves the performance especially for users who are using mobile devices to view the page. Besides, the file format of the icons in figure 13 is SVG which is a scalable vector format that can ensure the quality of the icons. Moreover, SVG can reduce the amount of requested data from the server which is considered as an improvement in performance for the page in overall.

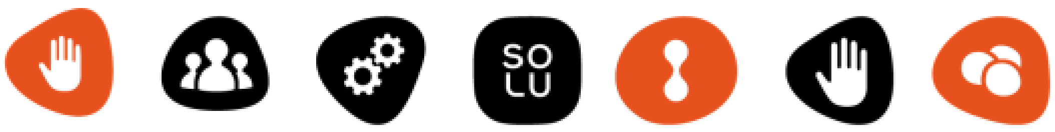


Figure 13. Using vector format to ensure quality and to improve performance.

The page was ready in three days and the key information on the page is integrity to all users regardless of the devices since the contents are displayed responsively by using the components of Bootstrap 3. However, there are minor things that could be improved in the project. For example, the content in the CSS file has not been optimized since there are many unusable classes. Even though the page works mostly on every device, there are scenarios that some components do not display properly. For example, in figure 14, the elements in the top navigation start to display unsystematically when the width of a viewport ranges from 772 pixels to 960 pixels.

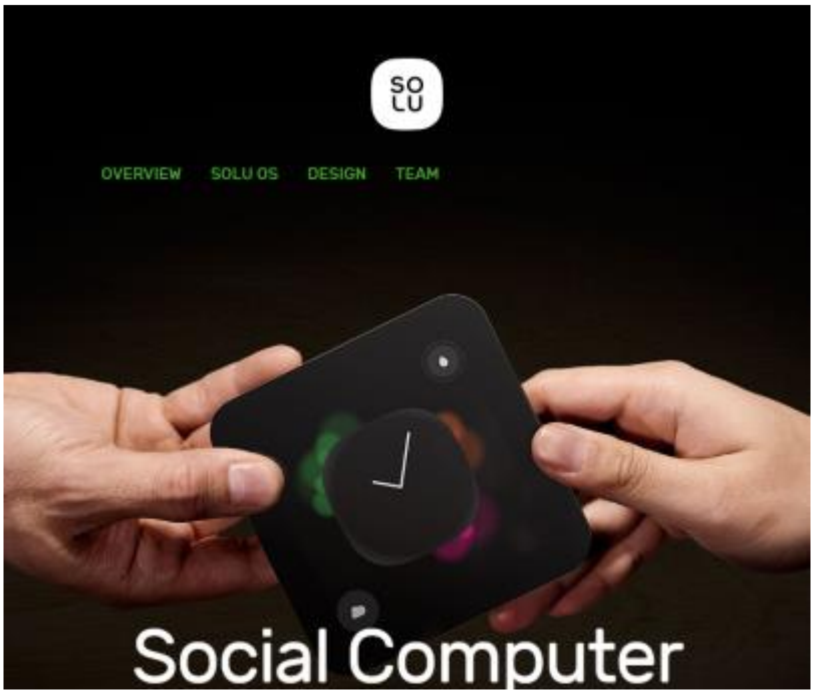


Figure 14. Broken navigation when the viewport is 900 pixels wide.

5 Conclusion

This study aims to analyse key principles and patterns that are useful in Responsive Web Design. The reason is to improve the overall process of designing and developing web pages in client projects. Mobile-first is a modern approach to creating responsive web pages effectively since it suggests prioritizing building all the components and interactions for mobile devices first. Using relative units instead of static units in CSS can ensure all the components will be displayed correctly regardless of displayed devices. Besides, there are many ready-made frameworks that allows developers to build responsive web pages faster.

To understand how key principles and patterns in Responsive Web Design are used in real projects, this research provides 2 case studies which were conducted while building websites for Finnair and Solu Machines. In Finnair's case study, a simple web page was built for Finnair using the mobile-first paradigm. All the components on the web page were created to be compatible with mobile devices first. In the following case, the landing page of Solu Machines was built using Twitter Bootstrap. In general, there are significant improvements in creating responsive web pages. Firstly, all the basic components of the web pages were considered from the perspective of mobile devices from the beginning of every project which reduces time and efforts when scaling to larger screens. Moreover, the usability of all web pages is consistent regardless of the device. Furthermore, the duration of each project is shorter compared to past projects that did not apply proper methods in Responsive Web Design.

There are issues which can be improved in the future for each project. For example, even though all images displayed on the web pages are scalable across all devices, sending the largest images to users is a speed issue for users with a weak data network. The author believes the process of feeding images from a server could be improved by providing correct images with the proper dimension based on the viewing device. Additionally, the pages are not optimized for all the available devices on the market.

In conclusion, as the amount of handheld devices will increase, Responsive Web Design is an essential part of every web design project. The process of applying principles and patterns to projects can be always optimal due to the constant evolving of the web technologies.

References

1 Zeldman J. Understanding Web Design [online]. A List Apart; November, 2007. URL: https://alistapart.com/article/understandingwebdesign. Accessed 16 January, 2017.

2 Calore M. April 22, 1993: Mosaic Browser Lights Up Web with Color, Creativity [online]. Wired; April, 2010.
URL: https://www.wired.com/2010/04/0422mosaic-web-browser/. Accessed 16 January, 2017.

3 Seigel D. Creating Killer Web Sites: The Art of Third-Generation Site Design. IN, USA. Hayden Books; 1997.

4 Lie H and Bos B . Cascading Style Sheets, Designing for The Web. Boston, MA. Addison Wesley; 1999.

5 Wroblewski L. Mobile First. New York, New York. Jeffrey Zeldman; 2011.

6 Marcotte E. Responsive Web Design. New York, New York. Jeffrey Zeldman; 2011.

7 Jehl S. Responsible Web Design. New York, New York. Jeffrey Zeldman; 2014.

8 Google. Material Design [online]. Google; 2017.
URL: https://material.io/guidelines/components/buttons.html#buttons-style. Accessed 15 November, 2016.

9 Mozilla Network Developer. CSS Values and Units [online]. Mozilla Network Developer; November, 2016.
URL:https://developer.mozilla.org/en/US/docs/Learn/CSS/Introduction_to_CSS/Values_and_units. Accessed January, 2017.

10 Marcotte E. Responsive Design: Patterns and Principles. New York, New York. Jeffrey Zeldman; 2015.

11 11. Disney. Disney Homepage. Disney; 2016.
URL: http://disney.fi/. Accessed November, 2016.

12 Bootstrap. History of Bootstrap [online]. Bootstrap; 2016.
URL: http://getbootstrap.com/about/. Accessed November, 2016.

13 Bootstrap. Bootstrap: CSS [online]. Bootstrap Documentation; 2016.
URL: http://getbootstrap.com/css/#grid. Accessed November, 2016.