Adaptation of an online course from desktop to mobile browsers

TEODORO ORLOW WEY
Adaptation of an online course from desktop to mobile browsers

TEODORO ORLOW WEY

Master’s Thesis at KTH
Supervisors: Prof. Anders Västberg, KTH Royal Institute of Technology,
David Tikka, CleverLearning AB
Examiner: Prof. Fredrik Kilander, KTH Royal Institute of Technology

TRITA-ICT-EX-2016:197
Abstract

This project investigates approaches for adapting desktop e-learning web pages to mobile devices using principles of Responsive Web Design. It gives a brief introduction on usability and responsiveness and how these concepts are important when developing online courses to be accessed by users in different devices. It explains an online course structure, its chapters and lessons, and how it is generated from a XML specification to HTML. After that, 3 different course web pages for desktop are optimized to be used in mobile browsers. The implementation section explains how the pages behave in smaller screens. In the conclusion, it is shown how some of the web pages design are not ideal for the mobile user and how to improve them in the future.
Sammanfattning

Acknowledgements

First of all, I must thank my parents, Natalia and Carlos, for all the love and support they gave me during this journey. Without you this wouldn’t be possible, I will be eternally grateful for all the help I received and will never forget how patient and caring you were. I’d like to thank everyone in my family, located so distant from Sweden but very close to my heart.

Thanks to Maria for the love and support given to me during this part of my life. To all my friends: Thanks for the great time shared with me during these two amazing years in Sweden. You were a very important part of my life here and will certainly be part of my memories forever. Furthermore, thanks to the CleverLearning team: David, Henrik and Peter. I appreciated all the good times and working experience.

Finally, I have to thank KTH and all the tutors that helped me fulfill this goal: May-Britt Eklund Larsson, who helped me even before I arrived in Sweden and was always very kind; The KTH professors and assistants, for not only teaching me but also encouraging me to be a better engineer and recognizing my views and opinions. Thank you prof. Anders Västberg and prof. Fredrik Kilander for the thesis guidance. I’m very proud of being part of KTH alumni.

Stockholm, 2016

Teodoro Orlow Wey
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>iii</td>
</tr>
<tr>
<td>Sammanfattning</td>
<td>v</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>vii</td>
</tr>
<tr>
<td>Contents</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xii</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xiii</td>
</tr>
</tbody>
</table>

## 1 Introduction
1.1 Problem .................................................. 16
1.2 Problem Statement ....................................... 16
1.3 Purpose .................................................. 16
1.4 Goal .................................................... 16
1.5 Methods and Methodologies ............................... 17
   1.5.1 Methods ........................................... 17
   1.5.2 Methodology ....................................... 17
1.6 Literature Study ......................................... 19
1.7 Benefits, Ethics and Sustainability .................... 21
1.8 Delimitations .............................................. 21
1.9 Thesis Organization ..................................... 22

## 2 Background
2.1 Responsive Web Design .................................... 23
2.2 User Interface ............................................. 24
2.3 Mobile Learning ........................................... 25
2.4 Mobile Web Browsing ....................................... 29
2.5 Front-end Technologies .................................... 30
   2.5.1 XML ................................................ 30
# Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM</td>
<td>Association for Computing Machinery</td>
</tr>
<tr>
<td>ADL</td>
<td>Advanced Distributed Learning</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets</td>
</tr>
<tr>
<td>DTD</td>
<td>Document Type Definitions</td>
</tr>
<tr>
<td>HCI</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>HTML</td>
<td>Hyper Text Markup Language</td>
</tr>
<tr>
<td>LMS</td>
<td>Learning Management System</td>
</tr>
<tr>
<td>MVC</td>
<td>Model-View-Controller</td>
</tr>
<tr>
<td>OOP</td>
<td>Object-oriented programming</td>
</tr>
<tr>
<td>RWD</td>
<td>Responsive Web Design</td>
</tr>
<tr>
<td>SCORM</td>
<td>Sharable Content Object Reference Model</td>
</tr>
<tr>
<td>SGML</td>
<td>Standard Generalized Markup Language</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Source</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Smartphone touchscreen displays in different positions.</td>
<td>Handbook of Mobile Teaching and Learning, 2015 [23]</td>
<td>25</td>
</tr>
<tr>
<td>2.2</td>
<td>Devices used for learning by percentage of total mobile devices.</td>
<td>ADL Mobile Learning Survey Report, 2013 [24]</td>
<td>26</td>
</tr>
<tr>
<td>2.3</td>
<td>The Two-Way Data Binding relation diagram in AngularJS.</td>
<td>AngularJS: Developer Guide [39]</td>
<td>32</td>
</tr>
<tr>
<td>3.1</td>
<td>Course page architecture</td>
<td></td>
<td>36</td>
</tr>
<tr>
<td>3.2</td>
<td>Non-responsive Introduction page in a desktop screen</td>
<td></td>
<td>38</td>
</tr>
<tr>
<td>3.3</td>
<td>Non-responsive Introduction page in a mobile device screen</td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>3.4</td>
<td>Responsive Introduction page from the bookstore course loaded in iPhone 6 Plus</td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>3.5</td>
<td>Non-responsive Hotspot page in a desktop screen</td>
<td></td>
<td>41</td>
</tr>
<tr>
<td>3.6</td>
<td>Non-responsive Hotspot page in a mobile device screen</td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>3.7</td>
<td>Hotspot page with responsive background image element and no responsive buttons in a iPhone 6 Plus screen</td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>3.8</td>
<td>The responsive Hotspot page in a iPhone 6 Plus browser, in portrait mode</td>
<td></td>
<td>46</td>
</tr>
<tr>
<td>3.9</td>
<td>A responsive Hotspot page in a different chapter of the same course.</td>
<td></td>
<td>47</td>
</tr>
<tr>
<td>3.10</td>
<td>A responsive Hotspot page in a different chapter of the same course.</td>
<td></td>
<td>48</td>
</tr>
<tr>
<td>3.11</td>
<td>A responsive Hotspot page accessed from an iPad browser, in landscape mode</td>
<td></td>
<td>49</td>
</tr>
<tr>
<td>3.12</td>
<td>Non-responsive Drag and Drop page in a desktop screen</td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>3.13</td>
<td>A tentative design for a responsive Drag and Drop page in a mobile screen</td>
<td></td>
<td>52</td>
</tr>
</tbody>
</table>
List of Tables

2.1 List of mobile device affordances. (Source: Handbook of Mobile Teaching and Learning, 2015)  [23]  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27
Chapter 1

Introduction

Good usability and interactive interfaces have pushed developers to begin using responsive web design techniques, increasing the usefulness of the websites accessed from mobile systems while maintaining the desktop website content [1]. These special conditions are crucial when building e-learning web pages, and are investigated throughout this project.

When designing an adapted version of a desktop website to be accessed on mobile systems, the layout characteristics of the web pages have to be changed. The coordinate system between both platforms is different, problems appear and need to be fixed to conform to the new environment.

The smaller screen layout issue has been explored by different web development companies and designers in the last few years [2][3][4]. One of the techniques that address this problem is Responsive Web Design.

Responsive Web Design (RWD) is a set of principles that has been proposed as way to make websites usable to the different devices that access them. The usability of interactive sites with learning content can be improved so that the features of regular desktop are displayed in the smaller screens without compromising the delivery of the courses.

The aim of the degree project was to investigate the usability issues of the mobile page of online courses. The courses developed by the company Clev erLearning AB are expected to have the appropriate responsiveness regarding the screen size of the user device. The following front-end technologies were involved in the investigation of the project: XML (EXTensible Markup Language) [5], HTML (Hyper Text Markup Language) [6], CSS (Cascading Style Sheets) [7], JavaScript [8], the jQuery library [9] and AngularJS framework [10] tools.
CHAPTER 1. INTRODUCTION

1.1 Problem

It is important to adapt the web pages when transitioning a website from desktop to mobile. The screen size and the lack of cursor are the main constraints in mobile web pages. The consequence is that interactive pages don’t function well on mobile screens. The size of the buttons decrease in an inappropriate way. The icons that are draggable in a desktop screen are not easily manipulated in a smartphone screen. This results in situations like scrolling the page on the small mobile screen and zooming, which becomes impractical for the learner.

1.2 Problem Statement

The courses developed by CleverLearning don’t have a page designed specifically to be used in mobile devices. The present thesis examined the challenges faced by web developers when designing an e-learning web page for multiple devices and proposes to answer the question: What are the alternative ways of displaying a web page on mobile devices, so that it improves the usability and learning experience?

1.3 Purpose

The system developed by the company works with a static web page that loads in the following type of devices: desktop, smartphone and tablet. Instead of having a particular page to display, there must be different pages for each device. This is a demand from the clients, because they use the courses for training purposes and wish to reach the users that are geographically dispersed more efficiently. Thus, they need training courses better adapted to mobile devices.

1.4 Goal

The goal of the thesis project is to show techniques in the field of web development to build websites according to the principles of Responsive Web Design in the context of a multi-device market. The evolution of browsers,
smartphones and Internet connectivity allow for users to access websites without having to download mobile native applications. This versatility should be explored, and adapting usable and interactive online courses to be loaded in mobile browsers can be useful for learning purposes.

1.5 Methods and Methodologies

1.5.1 Methods

The project applied a qualitative method of research. Since the project focused on web design techniques and how it affects people's experiences (usability), it used interpretative and critical assumptions to analyze the results. The research methods used are descriptive (it investigates and explains the characteristics of using responsive pages in mobile systems), conceptual and, most importantly, analytical.

The analytical method involves critical thinking and decision making towards design, which is the purpose of the degree project. The conceptual method is also used in the project because it is based on previous literature concepts.

Regarding the research methodology, the action research strategy is the most suitable for this project because it focuses on solving a practical problem and observing the results. This process was iterated until the best possible design for the system was reached.

1.5.2 Methodology

The project consisted of the tasks explained below. These tasks are not sequential, they include different aspects of what is necessary to complete the project work.

Literature survey

During the beginning of the project work, the focus was on finding material to understand the project subject. Books, papers, and articles on the subject were gathered from databases. Initially, 2 databases were used: KTH electronic
library \[12\] and KTH Primo \[13\]. They were used to find material by searching with keywords like: 'Web Development', 'Responsive Design', 'User Interface' and 'Mobile Browsing'.

After the references were found, they were saved in a digital directory named 'References'. Inside this directory 2 folders were created: 'Books' and 'Articles'. These folders included the references books and research articles, divided by the following topics related to the thesis work, listed below:

- Drag and Drop
- E-Learning
- Mobile Browsing
- Responsive Design
- User Interface
- XML

This step was fundamental to organize the work by gathering what could be important in the writing of the present thesis. As the references were collected, broader search was performed using popular search engines like Google to see if other work and ideas were found. Other databases that were used were the ACM (Association for Computing Machinery) and Wiley libraries.

After reading the Abstract and References sections of the selected papers, they were read and highlighted in the most important sections. These highlights were reviewed and included in the thesis if they were considered relevant for the thesis.

**Company’s software**

Understanding of the source code which the company uses to develop the clients projects was needed before any implementation was made. After learning the tools used in the company’s software, a test environment separate from the production branch of the code was used for the development of the mobile interface.

**Development and evaluation**

The project development was agreed in a series of meeting with David Tikka, the supervisor of the thesis project and owner of CleverLearning AB. During the first interview in the company, it was decided that the main technologies to
be used in the thesis project were decided JavaScript \cite{8}, AngularJS \cite{10} and jQuery \cite{9}. The following weeks were devoted to understand these technologies to apply them in the company’s projects.

After two months of training and study of the technologies, the design of simple mobile web pages started. CleverLearning has a development team in India, which gave full support during the progress of the development. If needed, the team in India was reached by call or email to help understanding the problems faced or to fix some technical issue. The two web pages to be worked on and improved as a goal for the thesis were the Hotspot page and the Drag and Drop page (please refer to Chapter 3 - Design and Implementation for more information). When each of them were finished, an evaluation was conducted by David Tikka.

For the evaluation, a walk-through of the web pages was conducted, using the company environment consisting of a web server. It showed the page live on a mobile device running as a fully functional online course. David would point the problems he found in the working design according to the company goals, and if the work was satisfactory the project moved to the next page implementation. Then, a conclusion was written about the key problems faced in the implementation and suggestions to improve the pages of the application.

Report and presentation

This was the final task for the degree project. After finishing the implementation, a report and a presentation were prepared according to the guidelines of KTH, the ICT School and the department of Communication Systems (CoS), followed by a public oral seminar.

1.6 Literature Study

To understand what had been researched and implemented previously in the field of this thesis project, a search for background work was made. For the description of the process, please refer to the “Literature Survey” in Section 1.5.2. The present section is going to discuss this selected background work.

The paper titled “From elearning to mlearning: the effectiveness of mobile course delivery”, by U.S. Navy researcher Jason Haag, was an important source of material for the present thesis. It’s a study on why to build versions of e-learning courses for mobile devices. In his paper he explained how the mandatory courses taught in the military needed the be coverted to be sup-
CHAPTER 1. INTRODUCTION

ported by “a wide range of mobile device platforms” [14]. The approach he used applied server-side detection to deliver the mobile version.

Haag’s paper that motivated the thesis work was the preliminary survey about mobile-friendly course versions. He asked the participants whether their organizations offered it, and 8% said the mobile-friendly versions were available [14]. It was interesting to have this perspective on the usage of mobile devices to deliver e-learning courses, because the survey shows how it is important to work towards an improvement. The section “Benefits of Mobile Course Delivery” of the paper explains that survey participants “believe that mobile course delivery provides many benefits” [14]. They refer to time management, touch screen interactivity and convenience as good features of mobile devices in e-learning.

Some problems were found in the mobile courses according to the survey participants. Interfaces problems were faced by the users, such as design glitches related to the rotation to landscape orientation, issues with touch screen buttons and nonfunctional videos [14]. All of them were faced during the thesis work.

The first reference about Responsive Web Design that was found in the literature survey was called “Responsive design for transaction banking - a responsible approach” [15]. It was saved and labeled as important since it discusses an “app that is ported to a tablet and smartphone context from a desktop based web app”. Its main focus is on building native applications to solve the issue of developing different experiences for each kind of screen and device. Thus it was not about to building web pages like the thesis project but mobile apps, so the approach it described was left out of the implementation of the thesis.

Another article about Responsive Web Design that was found is “One site fits all: responsive web design”, by Nancy R. Glassman and Phil Shen. It explains the design practices for creating responsive web pages. It was one of the first articles read during the making of the thesis work. It was useful because it describes the design process briefly and clearly. The section “The nuts and bolts of responsive web design” explains how to use web technologies (HTML5, CSS3 and Javascript) to define flexible images and media queries for the responsive design implementation [16]. These design details are used in the implementation and are referred to in the Section 3.2 (Techniques) of the thesis.

Some articles were not as important for the thesis as those previously cited in this section. They were found and and saved as part of the literature survey since they had some keywords related to the thesis project. For example, the articles “Requirements of Mobile Device’s User Interface in Social Networking” [17] and “The Importance of Mobile Interface Icons on User Interaction” [18]
were read since they refer to user interface issues in mobile devices, but were not included in the discussion of the thesis. They reflect on poor mobile interface and but the discussion on them was too superficial for the purpose of the project.

During the making of the project Google Developers website was a source of good practices for mobile web design. The Google Web Fundamentals page is a reference for this thesis work [4]. It recommends basic settings and configurations in the HTML and CSS code to turn a regular web page into a responsive web page.

### 1.7 Benefits, Ethics and Sustainability

The content of this thesis project was web development and involved designing an interface so that users can have access to mobile applications with improved usability. The project main focus was on the benefits of good design in an e-learning environment. The improvement of the CleverLearning online courses meant better usability for the learner, which also benefited the customers who chose to use the company’s projects for teaching purposes. By developing this software project, the company contributes by bringing digital solutions to the learning market. With the technology used to create the product and the means to access the content using mobile devices, CleverLearning provides an alternative to, for example, physical books and lectures, making the learning experience simpler.

### 1.8 Delimitations

The aim of the thesis was not to implement solutions based on surveys or opinion polls. A survey to collect users feedback on the web pages implemented was not performed in this project, since it was not included neither in the project, due to time limitations, nor the company plans.

The scope of the work was within web development for mobile devices and Responsive Web Design. It strictly followed the CleverLearning specifications and technologies, together with chosen web design principles that were considered important tools to improve usability and responsiveness of web pages. In other words, there was a model to be followed for implementing of the project,
which was a system that can identify the device used and adapt the layout and content presentation of online based courses.

There was a commitment to maintain the company’s proprietary code out of the discussion of the thesis. In the Design and Implementation part, the code parts owned by CleverLearning that are confidential are not presented and the explanation is reduced to a high level view of the system.

1.9 Thesis Organization

The present thesis consists of four parts, namely “Introduction”, “Background”, “Design and Implementation” and “Conclusion”.

The first part, “Introduction”, is divided in five sections (including this one). It starts with the “Problem Statement”, which describes the problem area relevant for the study, the reasons for examining this particular problem and the statement that explains what will need to be resolved. “Methods and Methodologies” explains the methods and methodologies of research used to investigate the problem. The next sections “Benefits, Ethics and Sustainability” and “Delimitations” state the outcomes of the thesis results in the area studied and the limits of the project research.

The second part is the “Background” chapter. It gives a brief review of the areas of study and technologies in the context of the research.

Next part is “Design and Implementation”. It presents the details of the project practical work, specifically the technologies used, code explanation and the results achieved during the process.

Finally there is the “Conclusion” part. It discusses the project outcome in the “Summary” section and the possible improvements in “Future Work”.

22
Chapter 2

Background

2.1 Responsive Web Design

The main focus of the project was on how to build responsive pages that maintain usability both on desktop and mobile, making the necessary adjustments in the layout to fit the screen size of different devices, such as smartphones and tablets.

When adapting a mobile page from a implemented desktop version, the steps to change the latter depend on what type of page the designer or developer is working on. The information that is displayed to the user is very important, because is not always a case of creating a new layout, but on how the different pages would be perceived by a mobile user. A good discussion about was found in the paper “Perceived Usability, Attractiveness and Intuitiveness of Responsive Mobile Tourism Websites: A User Experience Study” [1], by Groth and Haslwanter. Their study explains the different user experiences when trying two touristic websites on desktop and mobile.

The paper explored three main characteristics of a touristic website: usability, attractiveness, intuitiveness [1]. This thesis project, although not focused on user experience in this kind of market (touristic information websites), explored the latter three characteristics on Responsive Web Design, with more emphasis on the usability. Its goal is to find the most convenient way of adapting a website to the smaller screen, both mobile or tablet, without compromising the content of. In the paper called “Being Responsive” [19], the author Robert Fox discussed if it is good or bad to a business to have different websites versions, and what focus the developer should pursue when doing a website. A relevant point is made in the following sentence from Fox:

“...in many ways, success is measured by the visceral impact that a
web site can produce on the information consumer. Responsive design, however, makes an important assumption about the function of the internet as a vehicle for information discovery and retrieval. The focus is on transparency rather than glitz, and this is important because in reality, there are limited contexts in which people expect graphical richness when using the internet, such as in entertainment contexts, marketing sites, and cultural expositions.”

The focus, according to Fox, should be on the information that the page contains, on how it should be visualized without enhancements on the design or “glitz”, as he called it. This was the approach taken in the implementation of the project. The web pages that were worked on CleverLearning for this thesis didn’t actually need to be reinvented, they only had to be adapted for mobile screens so that the content could be more visible for smaller screens.

### 2.2 User Interface

The usability of a mobile page was a big concern in this project since the goal was to convert a page from the desktop screen to a new, redesigned page that can conform to the limitations of a smaller screen size, expose all the information present in the original version, and still maintain the buttons and links that make the page interactive. For example, since smartphones don’t have cursors or pointers, the page had to be clickable using the touchscreen feature, but with the condition that they are big and well spaced, and that the user could actually interact properly with the content.

On the paper “Usability of web interfaces on mobile devices”, Wessels, Purvis and Rahman explained good practices which were used to implement the web pages during the thesis work like, for example, the similarity between the desktop and mobile interfaces. In CleverLearning, one of the pages had buttons that, when pressed, displayed a pop up window with information. One of the concerns was how to maintain these buttons and overall interface as similar as possible to the regular desktop version.

Bringing educational websites to mobile devices that feature touchscreen interfaces allow more interaction between the student and the course content. The experience that HCI (Human Computer Interaction) provides is useful because it is appealing for the learner. Many features of the new devices were considered during the design decisions of this thesis. For example, instant
feedback, quizzes, and other sorts of interactive pages were included in the web pages worked in CleverLearning. Also, many tablets and smartphones have all the enhancements needed to build an interactive user interface. Touchscreen capabilities, different screen sizes, holding positions influence how a student can interact with a course page.

Figure 2.1. Smartphone touchscreen displays in different positions. (Source: Handbook of Mobile Teaching and Learning, 2015) [23]

2.3 Mobile Learning

Mobile learning (or mLearning) deals with the future technologies that are going to support the learning experience of mobile users. It is described as:

“Leveraging ubiquitous mobile technology for the adoption or augmentation of knowledge, behaviors, or skills through education,
training, or performance support while the mobility of the learner may be independent of time, location, and space.”[24]

Many government institutions are maintaining study groups and standardizing mLearning, like the Advanced Distributed Learning (ADL)[25], an initiative created in 1997 by the The Department of Defense (DoD) Office of the Under Secretary of Defense for Personnel and Readiness (OUSD P&R). For example, ADL created the Sharable Content Object Reference Model (SCORM), which is responsible for the standardization and specification of e-learning management systems[26]. CleverLearning follows the SCORM specifications for designing the company’s online courses.

![Image of devices usage](image-url)

**Figure 2.2.** Devices used for learning by percentage of total mobile devices.
(Source: ADL Mobile Learning Survey Report, 2013) [24]

Studies by Malone and Lepper [27] and Ciampa [22] show how several
2.3. MOBILE LEARNING

aspects of mlearning (challenge, control, competition, cooperation, curiosity, and recognition) have good impact on the learning experience.

The educational material for mLearning must be ready for the constant evolution of the users’ devices. As explained in Section 2.2, the manufacturers of the mobile devices have created interfaces that are intuitive. During the development of the thesis project, the main mobile devices explored were smartphones and tablets. These 2 devices were considered the most relevant for the research because they gather the majority of users. Figure 2.2 shows a survey made by ADL with 831 participants. It shows that these devices are the most often used by the participants, with tablets as the main device (61%), followed by smartphones (29%) 

According to Haag and Berking [23], the design of mLearning projects must take into account the affordances that enable the learners to perform actions on mobile devices. These devices have a list of capabilities (Table 2.1) that provide the educational designers with possibilities to create mLearning projects appropriate for the mobile environment [23].

<table>
<thead>
<tr>
<th>Affordance for Mobile Learning</th>
<th>Device Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessing: On-demand access to information, courses, performance support or refresher knowledge. Examples: Search knowledge bases, job aids, reference, dictionary, Wikipedia, courses, voice search, social media.</td>
<td>touch screen, internet browser, connectivity, microphone</td>
</tr>
<tr>
<td>Augmenting: Overlaying still imagery, audio, or video over real world objects or setting in support of or during a contextual learning activity. Examples: Augmented reality, scavenger hunt, museum tours, language learning.</td>
<td>camera, GPS, internet, connectivity</td>
</tr>
<tr>
<td>Capturing (audio): Documenting or recording auditory content in support of or during a learning activity.</td>
<td>microphone, speakers, digital storage</td>
</tr>
<tr>
<td>Capturing (imagery or video): Documenting or recording visual content relevant to learning activity.</td>
<td>camera, microphone, digital storage</td>
</tr>
</tbody>
</table>

Table 2.1. List of mobile device affordances. (Source: Handbook of Mobile Teaching and Learning, 2015) [23]

27
Table 2.1. (Continued)

| Communicating (messaging): One-way, two-way or group messaging as part of an informal or formal learning activity. Examples: Group collaboration, instructor/student discussion and chat. | SMS  
MMS  
chat apps  
microphone |
| Communicating (voice): Two-way, or group discussion as part of an informal or formal learning activity. Examples: Group conference, meeting, focus group. | voice call  
voicemail  
speaker  
microphone |
| Contextualizing: Notifications and linked interactions sent by transmitters or tags attached to objects using proximity or location sensors to provide a context-aware or location-aware content in support of or as part of a learning activity. Examples: iBeacons, QR Codes, scavenger hunt, mobile tours, games, and interactive stories. | Bluetooth  
GPS  
NFC  
RFID  
Wi-Fi  
camera |
| eReading: Accessing and reading documents on multiple devices anytime, anywhere in support of or as part of a learning activity. | text zoom  
text highlighting  
notes |
| Media Playing: Accessing media anytime, anywhere in support of or as part of a learning activity. Examples: YouTube, Khan Academy, Webinars. | image  
video  
audio  
internet  
connectivity |
| Notifying / Reminding: Event triggers, instant reminders, and alerts that illicit immediate responses or deeper engagement with a learning activity. Examples: Spaced repetition/learning, flash cards, language learning. | connectivity  
touch screen  
push notification  
service  
calendar |

Table 2.1 describes Accessing, eReading and Media Playing, the required affordances that the web pages in the thesis project required. They involve touch screen, Internet browsing and connectivity, text notes, image and video.
2.4. MOBILE WEB BROWSING

files. These are the capabilities related to the architecture (Chapter 3) that CleverLearning built. They are explained below:

**Accessing**

The courses built by CleverLearning offer web-based training as a solution for companies that require their employees or clients to learn about a specific topic. Since the companies demand support for online browsing, the courses must be available on the Internet.

**eReading**

The support to access the courses in devices such as smartphones and tablets is one of the concerns of the thesis project. Another one is how the content is exposed. Displaying text and information on the web pages in an appropriate size and in a coherent way with the course is an important quality that allows the learner to have the same experience as in desktop browsers.

**Media Playing**

The same principle for eReading is applied for information in the form of images and videos. In Chapter 3, the implementation of the Hotspot page shows how the responsive image is an important quality when accessing the course on the mobile screen.

2.4 Mobile Web Browsing

The implementation of the mobile web page of the project required the study of the characteristics of the smartphones and tablets, among others, and how to browse the page that was going to be tailored for these platforms. There are many technical details to be investigated in this topic, such as the screen vertical or panoramic orientation, the use of a physical or screen keyboard, etc., and they all relate to the usability discussion of the previous section. Studies have shown that, when browsing websites on mobile phones, users can complete specific tasks faster with mobile tailored versions as opposed to a non-tailored mobile version [28].
CHAPTER 2. BACKGROUND

2.5 Front-end Technologies

This section explains the related technologies and skills that were needed during the modeling and designing of the thesis work. It refers to the engineering and technical knowledge required to implement the web pages for the thesis project. The following is a brief definition of XML, HTML, CSS and the JavaScript tools.

2.5.1 XML

XML (EXtensible Markup Language) is a markup language that extends the HTML by allowing custom tags that describe data. It is based on the Standard Generalized Markup Language (SGML), a set of syntax rules for general markup languages created by the ISO in 1986 [29].

The tags are defined to store and transport data that is displayed by HTML [30] [31]. The XML and HTML syntax are very similar. Both have elements, tags and attributes, but XML adds information in the tags names, providing how a document should be presented by the HTML. The HTML syntax then extracts all the information out of the XML and translates it to a presentation layout that is compatible to the web standards. The instructions for the XML grammar are defined by Document Type Definitions (DTDs) [32].

2.5.2 HTML/CSS

HTML and CSS are the main technologies used for web development today. They are responsible for the majority of web pages built, surpassing others like Adobe Flash [33] as the standard front end development technologies. HTML (HyperText Markup Language) is the markup language responsible for the DOM of each page. It defines the structure of the website. CSS (Cascading Style Sheets) is a set of style rules that creates the visual details of the page. It is responsible for the content design like colors, fonts, image sizes. It also adds interaction to the user. That interaction is obtained by writing programming scripts for each page in a language called JavaScript.
2.5. FRONT-END TECHNOLOGIES

2.5.3 JavaScript

JavaScript is a language that manages the interactions of a web page. It works together with HTML and CSS to make the website live by defining functions for elements in the page. Buttons, images, videos, inputs are managed by JavaScript. In JavaScript, it is possible to implement classes that inherit from other classes. The concept of multiple inheritance is common for code reuse and extensibility in classical OOP (Object-oriented programming) languages like Java and C++ when creating classes and objects. This represents a “is-a” relationship between the objects in a hierarchy of classes. But another approach to object orientation that JavaScript supports is named object composition. This represents a “has-a” relationship.

Stampit is a library of functions written in JavaScript that substitutes this use of class inheritance common to OOP and implements a “has-a” relationship. The Stampit function library was introduced by Eric Elliott as an example of how to use factory functions to create objects with reusable behaviours, creating a class-free program. Factory functions are a way to create new objects in JavaScript. Stamps are composable factory functions.

2.5.4 jQuery

jQuery is a client-side JavaScript library that is based on simple syntax to create complex interactions in the web page. It is used for: HTML and CSS manipulation, event handling, creating visual effects. It is the most used JavaScript library today, with 70.6% of all the websites making use of it (96.2% of the websites that use JavaScript).

2.5.5 AngularJS

AngularJS is a client-side JavaScript framework that creates a structure based on Model-View-Controller (MVC) architecture. It extends the HTML by introducing new markups, synchronizing JavaScript files, separating the application logic from the view and binding the data the user sends and receives to the web page. In this manner, AngularJS decentralizes the implementation and updates between JavaScript and the HTML.
Understanding AngularJS Two-Way Data Binding

The AngularJS framework introduces a software architectural pattern for web applications isolating the application logic from the user interface (commonly known as view), improving code readability (separating the program into distinct files) and reusability. The specific architecture of AngularJS is divided in the following components:

- **Controller**: Controls the flow of data in the application. Contains properties and functions to be used in the application.
- **Model**: The component responsible for managing the application data.
- **Template**: The HTML code which is compiled in the browser, creating a live View page.
- **View**: The page that is loaded in the browser, based on HTML Template.

This type of relationship is fundamental to the AngularJS Two-Way Data Binding feature [39]. Two-Way Data Binding is a form of automatic update between Model and View: The data predefined in the Model is loaded into the View (the web page) when the Template is compiled. If there is any change on the data in the Model, the View reflects this change, and vice-versa (Figure 2.3). This feature makes the loading and testing of the web applications faster, since the server-side application does not need to update the entire page, only what is changed in the Model or View.

![Figure 2.3. The Two-Way Data Binding relation diagram in AngularJS. (Source: AngularJS: Developer Guide) [39]](image-url)
Chapter 3
Design and Implementation

3.1 Overview

The design and implementation part of the project focuses on the building process of a functional web page that works in multiple devices. It addresses the usability issues of the Introduction page, the Hotspot page and the Drag-and-Drop page of the CleverLearning e-learning system. Using principles of Responsive Web Design, the page is explained with emphasis on how to adapt a page in order to make it flexible and usable for learners that log in the CleverLearning courses. This explanation focus on the XML and the HTML codes and the JavaScript solutions created to describe the new web pages. The details that are property of the CleverLearning system are put out of the discussion, as noted in Section 1.5 (Delimitations).

At CleverLearning, all courses are produced over a specific Learning Management System (LMS). During the thesis work, all the HTML page types contained in the LMS were provided in the form of a template and all the modified implementation is kept in a different location from the production code been used by the clients.

3.2 Techniques

During the making if this project, various concepts and techniques were used to implement the web pages. Besides using the languages and technologies already mentioned in Chapter 2 (Background), the following were important features taken into account in this implementation:
• **Make the HTML elements responsive:** A front-end framework is needed for the project web page to be responsive. It simplifies and standardizes the site structure giving it a template and applying the same layout settings.

• **Media queries:** A media query consists of the media type attribute \[40\]. It checks the device that is being used, and loads the specific rules depending on device. Since CSS3, the media queries can make conditional statements for making different adjustments in the content width or ratio, for example, as shown below:

```css
@media screen and (max-device-width: 480px) {
  /* Add the CSS rules and styles here */
}
```

• **Fluid grids:** For the responsive page to work on different devices, a fluid grid must be used by assigning relative units to the CSS rules \[41\]. Instead of the absolute units, the relative % symbol should be used.

• **Flexible images:** The images of the the web page are important elements that must be flexible. When a browser size is decreased by the user, or a page is loaded on smaller screen devices, the image can remain larger than the browser window or the actual device being used. A simplified CSS usage of flexible images or other elements is the following:

```css
img, object, video {
  max-width: 100%;
}
```

### 3.3 Architecture

The LMS consists of an e-learning based system of courses, containing the standards and conventions to build a course. The CleverLearning courses contain a number of chapters and a number of web pages in each chapter.
3.3. ARCHITECTURE

Each course has a linear structure of page types, with content based on the client’s desire. For example, a bookstore that maintains an e-learning course to train its employees may have multiple choice question pages, knowledge test pages; an organization with an e-learning course for its members can display information videos on a page and have a quiz page after that.

A course contains between 3 to 7 chapters, or more if needed. It could also be possible that a client company has a course that only consists of one chapter.

Here’s the main contents of a course:

- A limited number of web pages within each chapter, in most cases 3 to 20 pages.
- A user interface that consists of:
  - The course title;
  - The customer’s logotype;
  - The page content;
  - Forward and backward navigation buttons;
  - Links to popup windows, such as help and dictionary;
  - A progress bar that shows the user’s progress within each chapter;

The CleverLearning LMS is responsible for the user progress and results. The foundation of this proprietary LMS system is a XML page that is used to create and publish the courses. The XML contains the information needed to generate the website in HTML. Each page type has an HTML template that is used as basis for the clients courses and their needs, and this template is generated from the main XML file.

The LMS specification defines that each course must have it’s structure written in a main XML document as a standard for all projects. From this file it is possible to retrieve all the information about the page type, the text, the questions, and other media contained inside of it (images, video, sound).

For the course pages to work, the code is separated in files that are used to render the pages design and functionalities. The files are written in JavaScript and use the AngularJS framework to make the Two-Way Data Binding (as explained in Section 2.5.5) with the server or course database, as well as interactions between the user input and the page. Below are files necessary for each page type files:
CHAPTER 3. DESIGN AND IMPLEMENTATION

![Course page architecture diagram](image)

**Figure 3.1.** Course page architecture

- **pagetype.js (AngularJS Module file)** - This file defines and initializes the AngularJS application. It’s a container that holds the information on other parts of the application. [42]

- **pagetype.ctrl.js (AngularJS Controller)** - The Controller that manages the data is defined in this file. [43]

- **pagetype.drct.js (AngularJS Directive)** - The directive contains the structure to create or change the HTML elements in the page. [44]

The HTML template that process the XML above and create the page for desktop and mobile is:

- **pagetype.tpl.html (template of the HTML page for desktop)**

The main XML file contains the pages defined for the course to be build. The structure of the XML file is based on CleverLearning naming conventions. With the information contained in the XML document is possible to generate the course HTML pages to the desktop and mobile browsers. Each page type is written in HTML and loaded using the AngularJS framework to build web applications.

### 3.4 Page Implementation

For the implementation of the project web page, a series of principles were taken to reach the purpose of the thesis project, along with CleverLearning
own needs. The list of guidelines for the project implementation are listed below. They are enumerated below as steps so that it is clear which techniques were used in each of the pages to be implemented. Each of these pages are described in Sections 3.5.1 to 3.5.4. Please note not all of the steps below are taken into account in every page. They are not mandatory in every page, but are enumerated because they were necessary in different page scenarios.

- **Step 1: Make the web page ready to be responsive.**
  Using frameworks, meta tags and media queries setup the page to make it responsive to the device screen.

- **Step 2: Flexible layout.**
  The elements and layout should be flexible and not fixed. This is obtained by using CSS rules.

- **Step 3: Images and text must maintain legibility.**
  The change between different screen sizes make the element text and image details decrease, which make the information difficult for learner to understand. This has to be fixed by correcting the size of the elements.

- **Step 4: Easy to build.**
  The pages should be easy for the developer who is building the course page, since it’s being done manually.

### 3.4.1 Introduction page

In this section an non-responsive page that has an introduction video for the course is explained. The standard structure of the page is explained as follows:

In the top right corner there is a menu button to navigate through the lessons. Once it is clicked, the list of lessons shows up on the screen. The bottom of the page contains the progress bar. It shows the current status of the learner in the course. The learner can also navigate by pushing the backward or forward arrows located at the bar. Finally, inside is the actual page type with its content, depending on the page type. Figure 3.2 displays the first page of the sample online course of the project, an instructive course about a bookstore. It displays a video with introductory information about the course and the book company that it was built for. The implementation in the current and next sections omit the details about the company and other design choices that are not important for the project to achieve its goal, therefore the images are sketches and not the final product.
CHAPTER 3. DESIGN AND IMPLEMENTATION

Figure 3.2. Non-responsive Introduction page in a desktop screen

The page has only one element inside it, the video element, with the commands (play, pause, full screen, subtitles). If the page is loaded in a smaller device, the video doesn’t resize and remains bigger than the physical screen. This results in poor usability, and the learners won’t be able to watch the video in their device in the first page of the course, showing only a section of the video element, as seen in Figure 3.3.

Viewport meta element

The first measure to be taken is to apply Step 1. Setting the Viewport of the page. The Viewport is the area of the page that is visible by the user and varies according to the size of the screen. It was introduced by Apple in their Mobile Safari browser in 2007 [45]. To tell the browser that the page should be rendered to the width of the device, the Viewport <meta> tag that has to be inserted in the <head> section of the HTML pages:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

The content attribute sets two properties:

- **width=device-width** sets the content width to the width of the device
3.4. PAGE IMPLEMENTATION

Figure 3.3. Non-responsive Introduction page in a mobile device screen

- initial-scale=1.0 sets a 1:1 scale relation between the CSS pixels and Device-Independent Pixels (DIP).

CSS flexible width

Following Step 1, the implementation of Step 2 is explained below. The following change was made in CSS. First, add the properties width: 100\% and height: 100\% to the class named video. This makes the video element <div class="video"> flexible to the size of the screen.

The CSS rule for this class has to be changed to:
The flexible layout resulted in a page with the entire video and the action buttons displayed inside the screen, visible without having to scroll the page horizontally. Figure 3.4 is a screenshot of the same page in a iPhone 6 Plus, but with the responsive design.

Figure 3.4. Responsive Introduction page from the bookstore course loaded in iPhone 6 Plus.
3.4. PAGE IMPLEMENTATION

3.4.2 Hotspot page in portrait mode

This section describes the adaptation of the Hotspot page type of the bookstore project that conforms to a mobile device screen. The Hotspot page is an interactive page that allows the user to click different buttons located on the screen, launching popup pages to get information about a particular topic in the course lesson.

![Non-responsive Hotspot page in a desktop screen](image)

Figure 3.5. Non-responsive Hotspot page in a desktop screen

In Figure 3.5 it can be seen that the size of the screen has a lot of space to insert information without compromising the user experience. The Hotspot page contains different elements:

- the background image, shown in Figure 3.5 as the circles with drawings or pictures.
- the interactive buttons, which the learner has to push to see further information.
- the page title.

The background image and the buttons modifications are explained in this page implementation. They follow Step 1 and Step 2.

When the page is loaded on a mobile device, the page visualization is affected because it is designed as a static desktop version. The screen is
smaller and the page displayed is not resized to fit the screen and must be scrolled horizontally to show the information.

To be able to show the content in the mobile screen without having to zoom or scroll the page horizontally from left to right with the fingers. A flexible layout (Step 2) has to be implemented to resize the layout to fit the screen.

Figure 3.6. Non-responsive Hotspot page in a mobile device screen

Figure 3.6 shows the Hotspot page when accessed via a mobile device. The page remains unchanged, and the information contained on the page is not completely shown. Depending on the device, the user has to double-tap, pinch-to-zoom or scroll the screen in different parts to be able to see the rest of the content.

Viewport meta element

According to Step 1, the first measure to be taken is setting the Viewport of the page, as in the Introduction page (Section 3.5.1):
3.4. PAGE IMPLEMENTATION

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

The *content* attribute sets two properties:

- *width=device-width* sets the content width to the width of the device screen in terms of Device-Independent Pixels (DIP). [4]

- *initial-scale=1.0* sets a 1:1 scale relation between the CSS pixels and Device-Independent Pixels (DIP). [4]

**CSS flexible width**

The following changes were made in HTML and CSS as part of Step 2. First, add the property *width: 100\%* to the `<img>` tag with the background image id named `hotspot-background-image`, and disable the position property. This makes the image in the background flexible to the size of the screen. The `<img>` element containing the background image is the following:

```html
<img id="hotspot-background-image">
```

The CSS rules for this id have to be changed to:

```css
#hotspot-background-image {
  left: 0px;
  top: 30px;
  position: relative;
  width: 100%;
}
```

This implementation makes the background image resize with the browser window size. However the buttons are not yet responsive and maintain the same static position as if the page is loaded on a desktop. For example, in an iPhone 6 Plus (Figure 3.7), none of them appear except for one button, placed incorrectly:
Figure 3.7. Hotspot page with responsive background image element and no responsive buttons in a iPhone 6 Plus screen. The button appearing in the screen (a purple circle with a white ‘plus’ sign) is in the incorrect position.

Javascript code for responsive Hotspot buttons

The position of each button in the Hotspot page was implemented in the AngularJS directive source code. The directive tells the page elements how to behave. Therefore the solution is to change the jQuery methods that are applied to each button. This made the buttons flow together with layout (Step 2).

The first task was modify the JavaScript implementation in the AngularJS directive file, so that the page changes automatically, without having to manually change the XML. In this Hotspot page there are 10 interactive buttons. Each one is a different element in the page and displays a different popup screen when clicked. To make these buttons responsive, their position had to be defined using a jQuery method for each button. Below is the code section of the Hotspot directive that changes button number 1 in the page:
3.4. PAGE IMPLEMENTATION

$(hotspotButtons).each(function (index, value) {
    if(value.target === "button_number"){
        $(button).css({
            'top': value.x + '%',
            'left': value.y + '%
            'position': 'absolute',
            'cursor': 'pointer'
        });
    }
});

Here, target represents a button that is parsed from the XML and stored in the hotspotButtons variable. The top and left properties are set to be responsive by adding the value together with a % to the x and y values contained in the XML of the page. The values are extracted from the XML, collected by the controller and sent to the directive. This implementation made the button position change with the browser window, so when the visible area gets decreased, the buttons automatically move to be in the exact position relative to the background image. This was applied to each button inside the page using the same jQuery methods (.each() and .css()), making sure the position of the buttons match the right position on top of the background image.

All the buttons remained functional, since there was no modification in the implementation for the pressing and popping of the popup window of each button. The popup window is small enough and ready for the different screen sizes and didn’t show any differences between the old the new versions of the Hotspot page. The final result for this page can be seen in Figure 3.8.
Figure 3.8. The responsive Hotspot page in an iPhone 6 Plus browser, in portrait mode.
3.4. PAGE IMPLEMENTATION

Another page with less buttons is shown in the Figure 3.9 below, also loaded in an iPhone 6 Plus in portrait mode. Here, the 4 buttons are carefully placed to be exactly in the same place as when loaded in the desktop screen (Figure 3.10). The changes made the buttons responsive but the content of the page remained the same as in the original project created in the XML structure.

Figure 3.9. A responsive Hotspot page in a different chapter of the same course. This is a web page accessed from an iPhone 6 Plus browser, in portrait mode.
Figure 3.10. A responsive Hotspot page in a different chapter of the same course. This is a web page accessed from a desktop browser.
3.4. PAGE IMPLEMENTATION

3.4.3 Hotspot page in landscape mode

This page implementation follows Step 1 and Step 2, the same from Section 3.5.2. Figure 3.11 shows another Hotspot page from the same course. It also contains Hotspot buttons distributed along the background image, but is loaded in an iPad and shows an example of the device positioned in landscape mode. The background image is spread through the screen of the device, showing the content and the Hotspot buttons spaced through a large area. The big difference from the portrait mode from Figures 3.8 and 3.9 is the size of the Hotspot Buttons in relation to the background image.

Figure 3.11. A responsive Hotspot page accessed from an iPad browser, in landscape mode.
CHAPTER 3. DESIGN AND IMPLEMENTATION

3.4.4 Drag and Drop page

The Drag and Drop page type is another page from the CleverLearning system that was investigated for the thesis project to be optimized for mobile devices. This type of page consists of an exercise where the learner has to drag objects to a specific target. The page works properly when the course is loaded on a desktop, but like the Hotspot page explained in sections 3.5.2 and 3.5.3, is not ideal for mobile devices for the same issue of non-responsive design.

The page accessed in the desktop browser is shown in the Figure 3.12 below:

![Figure 3.12. Non-responsive Drag and Drop page in a desktop screen](image)

This page type presented a big challenge since it had the following list of requirements:

1. An arbitrary number of drag objects.
2. An arbitrary number of drop targets.
3. A background image that covers the whole content area.
4. A body text that is displayed above the background image.
3.4. PAGE IMPLEMENTATION

5. The drag objects and drop targets are images.

6. It should be possible to position the drag objects and drop targets on absolute coordinates within the content area.

7. There should be a button for submitting.

8. There should be a setting for whether or not all drag objects must be dragged.

9. There should be a setting for how the drag objects should land on the targets.

10. When submitted the answer should be evaluated, and the following feedback should be given:
    - Feedback if not all drag objects are placed, if necessary.
    - Feedback if all drag objects are placed right.
    - Feedback if some or all of the drag objects are placed wrong.

Requirements 1, 2 and 4 were the most important from the mobile design perspective, because they occupied most of the area inside the page. The draggable icons and the dropping targets in the Drag and Drop page are the interactive items of the page type. Therefore the implementation should be thought of optimizing the interface towards these elements. The first idea was to take responsive approach from Step 1.

**Responsive approach**

The first approach was to make the buttons responsive to the mobile screen as part of Step 1. An interface that was thought of was one where the objects get reduced according to the device and still be draggable by a finger. In the reference page “Size Tap Targets Appropriately” of Google Developers website about Responsive Web Design, the recommendation is that the developer must make important tap targets large enough, so that they are easy to press. The objects were made so that the minimum height was set to 48 CSS pixels. Following this guideline, a drawing of the mobile interface was made as an idea to how the web page should look on small smartphones and presented to CleverLearning to be evaluated. Figure 3.13 shows the resulting drawing.
This design approach focused on the usability of the page from the learner's perspective without replacing the original idea of the Drag and Drop page that is enumerated in the requirements list above. From Figure 3.13, it is seen that there is a small space left for a submit button (Requirement No.7). In Figure 3.14, we can see how the original Drag and Drop page would look like when loaded in a smartphone browser before the changes compared to the page.
3.4. PAGE IMPLEMENTATION

loaded after the implementation of the mockup design.

![Image of Drag and Drop page before and after implementation](image)

**Figure 3.14.** The original Drag and Drop page before (left) and after (right). The image on the right shows the implementation of the mockup design.

It can be seen from the above image that the drop targets are not fully shown in the leftmost image, but the constraints set by the mobile device appear when the number of objects increase. Suppose there are 10 or more draggable objects and its respective drop targets. Since the mobile screen is too small for that quantity of interactive objects, the learner will have to scroll down the page so that it finds the rest of the objects (if they remain with the same size of 48 CSS pixels minimum height). This will lead to problems because the simultaneous scrolling and dragging of the objects is not allowed. The touch screen functionality of the device cannot differentiate between the two.

The following is a simple scenario where this problem is encountered: A draggable object is held down with the finger in the screen. If the object’s target is not displayed in the screen, the user needs to scroll the web page until it finds it. The learner has to release the finger that is currently holding down the object. The learner then uses the same finger to scroll the page.
CHAPTER 3. DESIGN AND IMPLEMENTATION

This kind of scenario exemplifies the difficulties faced when adapting the Drag and Drop page for small screens in this project. It can become non-intuitive for the learner. If many draggable objects are included in the course page, the drop targets can be pushed down to the bottom of the page. Figure 3.14 shows how 3 objects can be enough to crowd the screen of a small screen device. They will only be visible as soon as the learner scrolls the page. The objects size cannot be decreased. This violates the legibility determined by Step 3.

Alternative page type approach

The learner can become unaware of the need to scroll down the page to reveal the drop targets that are part of the question in the page. This design had to be changed and the idea was refused by CleverLearning.

The next approach taken was to redefine the page for mobile devices as a different page type. To adapt the original Drag and Drop page for mobile screens was not practical, so a question type page was created, so that the mobile version included question that referred to the draggable objects and its drop targets. So if the Drag and Drop page contained three objects, the mobile version of this page would contain three statements (that referred to the text from the drop targets) and each of those a set of three answers (referring to the draggable objects). One of the answers is the correct one. Based on the same Drag and drop page from Figure 3.12 the design of the new question type page is shown on Figure 3.15.

This design was an appropriate alternative because the users don’t need to drag objects. They can simply scroll down if there are more questions in the questionnaire. The change in the implementation maintains the legibility required by Step 3. The idea was that course only loads the Drag and Drop page type in desktop. If the course is accessed in a mobile tablet or smartphone, the page loads as a Question page type. This however creates another code implementation for a single course page. The HTML has to be divided in two: one for Drag and Drop page type and another for Question page type.

The duplication of code and data explained above is undesirable for companies such as CleverLearning. The maintenance of the code can become a problem, making the courses harder to build, violating Step 4. Since no improvement was made if this design was implemented, this page was not re-designed. It was decided that so this page need further improvements, so no
Figure 3.15. A question type page: An alternative design for a responsive Drag and Drop page in a mobile screen.

work was made after the alternative Question page idea.
Chapter 4

Conclusion

4.1 Discussion of the implementation

The code that was introduced in Section 3.5.2 shows how the page is programmed to show all the buttons in a mobile screen without having to scroll or zoom in order to display information. The background image, the buttons and the popup screens are responsive to the screen size, so the users can interact with the page more easily whether they have a desktop, a smartphone or a tablet. Figure 4.1 shows the page loaded with the changes and how it looks when the browser window has the width decreased.

During the testing for the Hotspot page, the responsive buttons were originally implemented to be decreased in size so that they follow the size of the background image, making the design elements proportional. The results showed that the buttons’ size decreased in a way that is hard for the learner to press when using a smartphone. Their small size made it complicated for the touch screen in the device - which is already small - to sense the user’s finger. This resulted in worse user experience.

Following the guidelines from Google Developers page [46], the page buttons were made so that the positions became responsive to the browser window size, but their sizes remained unchanged. That way the learner could interact by easily pressing them as in the previous version. The other concern regarding is that the buttons should be proportional to the background image. Figure 3.5 shows that although the buttons are not precisely on top of their respective spots, this design detail don’t affect the overall design or usability of the page. The discussion of maintaining a distance between each Hotspot
button and their sizes had a positive impact that made this drawback of the buttons’ proportions irrelevant.

The image displayed by the Hotspot page was part of the fixed design and made flexible by applying the relative width rule to its CSS attribute (width: 100%). This was an easy adjustment for the Hotspot page, which has only one big background picture present in its design. Different websites that share this simple design also apply the same rule. Reidsma, for example, studied this design issue in libraries websites [47][48]. Comparing the same page loaded in a smartphone and on desktop, some differences were visible, mainly the space covered by the background image, which occupies less than 50% of the iPhone 6 Plus screen, compared to the desktop. The figures also show how the buttons maintained the same size in multiple devices, not disturbing learner usability of the page.

There are other Hotspot pages in this and other projects that were also adapted to have a responsive mobile interface. Since the changes were not meant to be made in the XML file, every other page has the same implementation for the position of the buttons as the Hotspot page explained in Sections 3.4.2 and 3.4.3. The only difference was in the position of the buttons in the other pages, and that depended on the XML description for the course. This XML definition is part of the production process of CleverLearning, and

Figure 4.1. The Hotspot page with responsive buttons in a decreased desktop browser.
because the company’s objective is to create the course data in XML beforehand during the project early stage, the developer can have less dependency on how the content was made and focus on parsing the data contained in the XML file and make the content behaviour change according to the page type that is used to load in the course web page.

4.2 Summary

This thesis project investigated how to implement better online courses by applying techniques of web development and design to achieve a responsive web page that fit multiple devices without making the learner struggle with bad usability. From the beginning of the project, the goal was to study and analyze the current status of the pages technical implementation. A lot of time was devoted to familiarizing with the tools to create new interactive web pages, having meetings with the developers about the architecture used by the company and self-study of JavaScript applications, AngularJS and jQuery, and responsive web design principles applied in HTML and CSS.

After this first phase, a literature study was started to gather information on the areas of E-learning, mobile web browsing, Responsive Web Design, user interface. They were organized in a separate folders, divided between papers, articles and books. After this was completed, the project implementation started by investigating the Hotspot page type from a course that has been designed in CleverLearning, and working on the responsiveness of the page. The debugging and testing were performed and the results were shown to David, the thesis supervisor. After the Hotspot page, the next objective was to adapt the Drag and Drop page for mobile. This page type was analyzed for a longer portion of the thesis project schedule. Two pages were designed, evaluated and refused. Since the results were not achieved for the design of the Drag and Drop page, and the schedule was delayed by the work, it was put as future work to be done after the conclusion of the thesis project.

The final phase of the project involved writing the thesis based on the project investigation and the information about good practices for building learning courses found on the literature study, along with evaluation by the supervisor and examiner of the thesis in KTH.
4.3 Future Work

This thesis project represented a sample of the work to be done to improve the usability of the web pages developed for e-learning courses. One future task is simplifying the XML specification for the courses to be generated. Other pages must be optimized for the mobile devices, but require improvements in the XML structure first. One of these page type is the Drag and Drop page.

The Drag and Drop page type has to be optimized to desktop and mobile devices. This type of page consists in exercises where the learner has to drag objects to a drop target. The page works properly when the course is loaded on a desktop, but as with the Hotspot page, is not applicable in mobile devices for the same issue of static design.

Finding a solution to this issue by applying a new design that can generate pages with different behaviour for desktop and mobile devices and when scaling down the window will have great impact in the resulting course look and feel, making the learning experience easier when accessing the courses in different platforms.
References


REFERENCES


REFERENCES


REFERENCES


[38] Google. *What is Angular?* URL: https://docs.angularjs.org/guide/introduction


