

Markus Ruottu

Tools and Technologies for Interactive Elements and SVG Animations in HTML5-based e-learning

Helsinki Metropolia University of Applied Sciences

Master of Engineering

Information Technology

Master's Thesis

1 Apr 2016

Author Title Number of Pages Date	Markus Ruottu Tools and Technologies for Interactive Elements and SVG Animations in HTML5-based e-learning 96 pages + 1 appendices April 1, 2016
Degree	Master of Engineering
Degree Programme	Information Technology
Specialisation option	Mobile Programming
Instructor(s)	Kari Salo, Principal Lecturer Eetu Tuomala, B.Eng, Senior Software Developer
<p>The research question that this thesis tries to answer is: What are the best tools and technologies to develop standardized activities and custom SVG animations for e-learning with HTML5 technologies taking into account the requirements and the restrictions of the mobile use and devices? The study tries to answer the question by dividing the studied topics into subcategories and selecting the tools and technologies best suited for each category and trying to get proof they are the most suitable choices with tests.</p> <p>The method to find the best tool was evaluating and scoring the tool candidates and their properties in each subcategory and comparing them against each other. The data used in the research was collected from a great number of sources, many of them in the Web.</p> <p>The research found tools that will benefit developing hotspots and carousel activities, and general or chart animations. Results for a character animation tool were thin. The background chapter of the study describes the technologies and best practices used in a modern e-learning course player.</p>	
Keywords	interaction, animation, activity, hot spots, carousel, chart, JavaScript, SVG, HTML5, e-learning, m-learning, responsive design, GSAP, Swiper, C3.js, qTip, player, devkit

Preface

As I started this study I had just moved back to HTML after being a Flash developer for almost a decade. So I basically needed to learn everything but the basics again. The technologies of Web development had developed enormously in that time. But I took the challenge assuming it would give me a crash course to catch up. However it did not end up being a crash course as it took me two and half years to finish it. The most difficult part of the study was to find all possible candidates for each specific task – how to be sure that the one, best matching the requirements, is still waiting to be found. I appreciate that my employer, MPS Prewrite, gave the necessary resources for me to finish the study, especially Eetu Tuomala and Juhani Mäkelä who I consider as my mentors. I also appreciate Kari Salo's supportive role as my supervisor and Metropolia for granting me extra time to finish the study. Special thanks go to my girlfriend who took the cooking responsibilities on the last months as I needed all the time available to finish the research on time.

In an airplane from Helsinki to New York on 5th of March, 2016, Markus Ruottu

Contents

Preface

Abstract

Table of Contents

Abbreviations

Glossary

1	Introduction	1
1.1	Scope of Study	1
1.2	Research Setting	2
1.3	Research Approach and Method	2
1.4	Research Design and Process	3
1.5	Research Question	4
1.6	Structure of Study	4
2	Background	5
2.1	Prewrite	5
2.2	E-learning	6
2.3	Mobile Web	7
2.4	Gimlet Flash Player	8
2.5	HTML5 vs. Flash	9
2.6	Gimlet HTML5 Player	10
2.7	Activities	10
2.8	Animations	11
2.8.1	Informative	12
2.8.2	Decorative	14
2.9	Client-side Requirements	15
2.10	HTML5	16
2.11	Best Practices	16
2.12	Responsive Web Design (RWD)	17
2.12.1	Responsive Layouts	19
2.12.2	Mobile First	21
2.12.3	Responsive Images	23
2.12.4	Problems	24
2.12.5	Future	25
2.13	SASS	25
2.14	MVVM and Knockout	26

2.15	Asynchronous Module Definition – AMD	28
2.16	Other Tools	29
2.17	Testing	30
2.18	SVG	30
2.19	SVG Tools	31
2.20	General HTML5 Supported Animation Technologies	33
2.21	Activity Tools	36
3	Method and Material	37
3.1	Methods	38
3.2	Process	39
3.3	Data	40
4	Tasks	42
4.1	Activities	42
4.2	Hotspots	43
4.3	Carousel	45
4.3.1	Requirements	47
4.3.2	Risks	48
4.3.3	Tools	49
4.4	Animations	51
4.5	General Animation	52
4.6	Charts Animation	54
4.7	Character Animation	59
5	Results and Analysis	61
5.1	Hotspots	61
5.2	Carousel	64
5.3	General Animation	67
5.4	Chart Animation	69
5.5	Character Animation	73
5.6	Discussion and Analysis	75
6	Summary	79
	References	81
	Appendix 1. GHP client requirements	

Abbreviations

CSS	Cascading Style Sheets
CMS	Content Management System
DOM	Document Object Model
HTML5	Hypertext Markup Language (version 5)
GFP	Gimlet Flash Player
GHP	Gimlet HTML5 Player
GSAP	GreenSock Animation Platform
IDE	Integrated Development Environment
IE	Internet Explorer
JS	JavaScript
LMS	Learning Management System
MVVM	Model View ViewModel
PC	Page Component
RWD	Responsive Web Design
SVG	Standard Vector Graphics
UI	User Interface
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WYSIWYG	What You See Is What You Get

Glossary

Activity	A term used for the standardized interactive elements in Gimlet HTML5 Player
Course Player	A customized course player based on Gimlet HTML5 Player DevKit
Devkit	Development kit
Gimlet	Prewrite's LMS/CMS software product family

Page Component	Page Component is a standardized activity or other feature used in Prewrite's e-learning courses
Prewrite	Finland's leading e-learning company
Tool	This term is used in a broad meaning when referring technologies, applications or JavaScript libraries that can be used as a tool to resolve a problem

1 Introduction

The key concept of this study is using thorough methodology to find the best available tools and technologies for specific purposes. First, study the subjects in general. Then define the requirements based on current knowledge. Then rate and evaluate the tools based on their properties and features against the requirements. Last make comparisons, qualify the best and test them in practice. The result of the study should provide the best tools for building specific activity types and animation types for e-learning courses.

1.1 Scope of Study

This thesis studies tools and techniques used to create interactive elements and animations used in e-learning. Interactive elements will be later referred to as activities for clarity. These tools and techniques can be HTML5 and CSS3 techniques and design patterns, JavaScript libraries and HTML5 Animation IDEs. These were selected as the research target because choosing and using the right tools, developers can save a lot of time and make projects more profitable. This research also takes account of responsive web design and best practices to utilize it, because it is an essential requirement for the results of the project. I decided to narrow the scope of the studied animation tools to SVG tools, because based in my experience; it is best suited for modern Web.

I could not find any studies related directly to more than one of my subjects at a time. So having the subjects together makes this study unique in the field. Other studies on the topics were on a more general level, from a different perspective or only about one tool or technology.

This thesis does not cover:

- Tools suitable for other page elements (or course elements) or exercises.
- Usability issues or user experience from the end users perspective with an exception of issues related to responsive design
- Issues related to customization of the activities or animations
- All tools and technologies related to HTML5 interactions and animations – only the specialized tools suited for each specific task in the thesis.
- Linux or Mac specific tools

E-learning in general is not the main subject of this study, but a framework where it happens. It will be touched on a general level and in the areas where it relates to the study topics.

1.2 Research Setting

This thesis was done for MPS Prewrite Oy (later just Prewrite). Prewrite is one of the leading e-learning companies in Finland. Until 2013, Prewrite used Adobe Flash as the client-side technology for e-learning courses. In 2007 Prewrite developed a development kit (devkit) to standardize common features and operation of the courses and to have a “clean” base to build the new courses on. The main reason was to minimize the customization work for each e-learning course.

Web has been adapting fast for the increasing use with mobile devices. E-learning also follows this track and so does Prewrite. Flash is not supported in mobile browsers. Therefore a new HTML5 based devkit was developed during 2013 and 2014 to answer to the growing need for mobile support. At the beginning Gimlet HTML5 Player (GHP) was very limited in features compared to its predecessor, Gimlet Flash Player (GFP). It still is, but not as much. It was decided that the development of GFP would end, and all efforts are put to the HTML5 devkit and it should eventually have at least the same features as GFP.

I was selected to do this research because I led the development GFP and I have the best knowledge of the standardized activities and animations used in Prewrite’s Flash-based courses. I also wanted and needed to learn HTML5 a lot better. Doing the research for the thesis was an effective way to accomplish that. My fairly poor or at least outdated HTML knowhow in the beginning of the research is the reason for great number of references found in the thesis.

1.3 Research Approach and Method

The research was divided to two phases. In the first phase the studied tools and technologies were gathered, evaluated, and compared against each other by their properties related to requirements derived from HTML5, mobile environment and experiences with GFP.

Tests and experiments were conducted to produce data missing from sources available. The best tools and technologies were selected based on comparisons.

The second phase was to get proof of concept. The selected tools were tested by developing two activities and three test animations. After implementation and the tests, the results were analysed and the final conclusions reported.

In phase 1 the idea was to find the right tools and technologies by comparing using evaluative approach. In phase 2 the qualified tools were tested by developing activities and an animation of each animation type for proof of concept. The research used both qualitative and quantitative methods. The qualitative method was used whenever information had to be converted to a numeric value so it could be used for comparison. The qualitative method was also used to analyse the proof of concept.

1.4 Research Design and Process

The study was started by defining the requirements for the tools and technologies. It was done by studying the properties, qualities and features of existing implementations in the Flash version of the course player. After the requirements were gathered, the theory was studied and best practices of HTML5 and the tools and technologies related to it, especially in the fields related to activities and animation. As studying the literature and other sources, the feature, property and quality information and data of the tools and technologies that could meet the requirements was gathered. The list of requirements with ones derived from HTML5, responsive web and mobile use was also completed.

The next step was to compare the tools and technologies and qualify the best. After having the best scored candidates of the comparisons – there was a need for proof of their applicability. Activities were implemented to final product and test animations were developed to get the proof of concept. Last the notes on the development experiences from were analysed with conclusions. The results of the study cannot be considered fully objective, because the evaluations are partly based on subjective estimations when data and other information had to be compared qualitatively.

1.5 Research Question

This study focuses on the HTML5 tools suitable for Prewrite's HTML5 development. The research question that thesis tries to answer is:

What are the best tools and technologies to develop standardized activities and custom SVG animations for e-learning with HTML5 technologies taking into account the requirements and the restrictions of the mobile use and devices?

The study tries to answer the question by selecting the tools and technologies and proving them as the best choices developing the cases presented in the thesis.

1.6 Structure of Study

This report is divided to 6 chapters. After Introduction comes the chapter with background information. The beginning of that chapter is about Gimlet HTML5 Player and the settings of the research. It explains the technical environment that sets the bounds to study topics and why the study was conducted. The chapter also covers the areas of the general technologies and some best practices related to the topics of the study. The information in this chapter is required to thoroughly understand the setting and the problem leading to the subject of one of the following chapters about the research tasks.

Before going to the tasks in-depth it was necessary to explain how the study was conducted, the methods used in the research, and describe how the data was gathered. This makes the chapter, Method and Material. The chapter Tasks is unconventionally its own chapter, because of the great number of tasks conducted in the study. It describes and defines each specific task topic, its requirements and the tools studied for the evaluations. The first half of the chapter goes through the tasks for the activities and the second part for the animation types.

After the Tasks, follows the Results and Analysis chapter, which reveals the results and tries to analyse them and draw conclusions from the findings. The last chapter, the Summary, sums up the study.

2 Background

This chapter covers the background material for the study. The first three quarters of the chapter introduce the operational environment for the research, to better understand the basis and the choices made in the research process. The last quarter covers SVG and animations and activity tools in general.

The most essential entity in this thesis is Gimlet HTML5 Player, later just GHP. It is what defines and connects every subject of the study. GHP is an e-learning course player development kit (devkit). It is used as a base when customized course players are built. Course players are web apps for presenting e-learning course content, provide interactive functions and send user's actions to server. By developing more features to devkit, it provides more standardized features to course players.

To understand what GHP is, it is useful to know something about the technical, operational and business environment it is used in. This chapter covers that framework.

2.1 Prewrite

Prewrite was founded 2004. It provides B2B change implementation and competence development, and is the market leader in online training in Finland. Clients are mostly big companies which results in distorted client browser distribution and versions compared to consumer systems. In the corporations Internet Explorer has significantly greater market share than in small companies or private use, and the browsers can be very old [1; 2].

Designing and developing e-learning courses has been Prewrite's main service from the start. Every step from planning to implementation to maintenance is done in house, apart from some special cases such as localization and global video streaming services.

Gimlet is Prewrite's product family including a learning management system (LMS) and learning community among others. GHP is part of Prewrite Gimlet product family.

2.2 E-learning

The history of e-learning starts from the 60's, but there is no specific time of birth or path of evolution recognized [3,1]. E-learning is learning material that is delivered in electronic format and typically in Internet-based technologies [4]. This definition is very broad and so is the variation of different types and formats of e-learning. One can often find even different variations of how the term is written.

E-learning is implemented in different ways in the sectors where it is used, sectors such as Business, Education, Training, and the Military sector. In global business environment e-learning focuses on content delivery and online course management. The aim is at increasing productivity and cost reduction. [3,1-2.] This is the very context where GHP is used.

There is a myth that one piece of software could provide the means to build e-learning projects [5]. Like web application nowadays, also GHP is developed using various tools, technologies and best practices.

E-learning in Prewrite and with GHP is mainly courses. They are typically made for companies to train their employees. A typical course consists of 20 to 35 pages with text, images, activities, animations, exercises and a final quiz. The most common page type includes only a title, a few paragraphs of text and an image. Standardized activities are added to a page to engage the users and to structure and fit additional information there. Find more about activities in Subsection 2.7. Custom activities can have more illustrative graphics and roles. Animations are used to bring graphics to life and more precisely to illustrate function, bind information to time, or just make a page more engaging with the attention to the movement. Learn more about animations in e-learning in Subsection 2.8.

Exercise pages have tasks such as multiple choice tasks, fill-in-the-blanks, or true-or-false. Their purpose is to train the user with the subject just studied. A quiz is frequently found at the end of the course. It typically consists of 5 to 10 questions that test how well the subjects of the content were understood.

The courses made for GHP course players can also be considered m-learning, but this definition is not as established. Some consider m-learning just e-learning shown in a mobile device, some consider it has certain requirements for its content and format too. Regardless, the aim is to create learning content supported or optimized for mobile use, and ena-

ble learning in a just-in-time, just-in-place dynamic. Use of m-learning should lead to increased productivity as it is accessible online anywhere and anytime. [4]

M-learning in general is best as compact and with little content [6,13], and often from a narrow topic. I would say compared to “normal” e-learning there should be less text and more activities, tests, and video in m-learning. In general, the GHP based courses can be considered m-learning in its broader meaning, and if needed, they can be customized to fall into the strict definition of the term too.

Although m-learning has been around for ten years, at least as a term [7], it is still in the early-adopter stage in corporate world [6,4]. One of the reasons to this is believed to be immature m-learning technology [6,5]. I must agree with the claim, but still state that the technology is getting mature for the next generation of m-learning. Bersin & Associates end their 2011 report of the state of m-learning in a thought that ‘m’ will not be needed for long, because mobile devices will be the normal way accessing electronic learning [6,50].

2.3 Mobile Web

Developer and author Stephen Hay states in his tweet, “There is no Mobile Web” [8]. He is absolutely right when looking from the end user’s perspective. But there is mobile use of Web and it sure has an impact when developing web sites. Even so that some of the sites have versions viewable in mobile devices, but yes it is still the same Web for all. The term is well established, so it is justified to use it. For the developers the mobile web is best practices, design patterns, and new technologies [9,38].

The experience accessing the Web when mobile or just with a mobile device is very different from the one with desktop or laptop. The smaller display with touch controls changes the browsing experience the most. A big issue in mobile use can be the sites with bad mobile support. This is often due to the small display or unsupported features. Also higher latency, slower networks, and less efficient hardware have an effect to the experience. [9,38-39]

Mobile use of Internet is increasing. It is already over 40% of the total use in Finland when tablet use is included in mobile use [10]. It is a bit less in Europe. The usages are shown in Figure 1. I would interpret the statistics so that if the long term trend does not change, mobile use will overtake desktop use in this decade – at least in Finland.

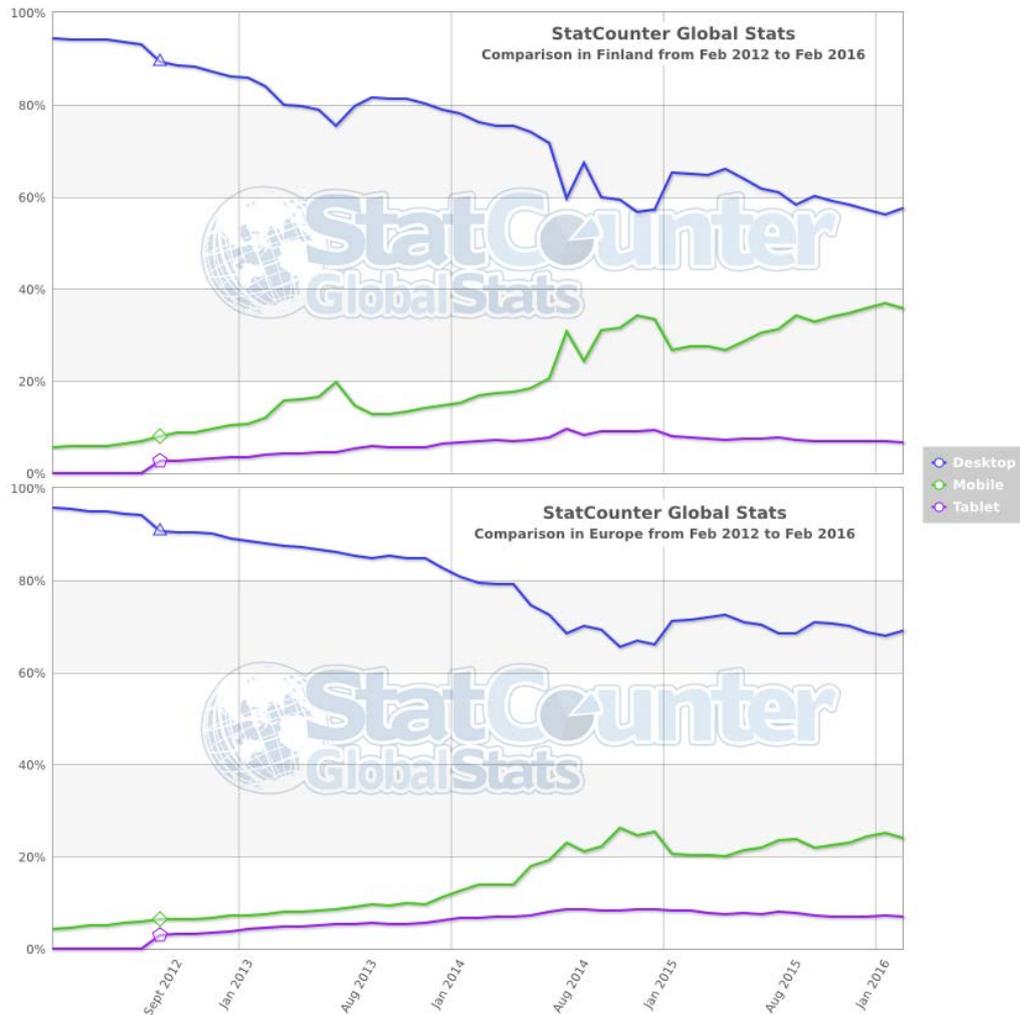


Figure 1. The graphs provided by StatCounter show how desktop, mobile, and laptop shares from all internet usage have developed in Finland (top) and Europe (bottom) [10].

Firtman (2013) claims that half of the developer's knowledge of the "desktop web" does not apply on mobile web [9,xvii]. Well, although the amount can be questioned, he has a point there that mobile development does require plenty of new know how about the usability patterns, best practices and mobile oriented technologies. Find more about a concept born for the mobile web in Subsection 2.12, Responsive Web Design (RWD).

2.4 Gimlet Flash Player

The history of the GHP starts when first version of its predecessor, Gimlet Flash Player (GFP), was developed in year 2007. GFP was developed on Adobe Flash technology, which was then very popular and often used as e-learning client technology.

In the versions 1.n GFP was not a devkit. New custom player was developed in every project from a copy of the last project. Version 2.0 was developed and used as a devkit. It had features supporting easy visual customizations, but had standardized features and UI. It also had a core, handling the connections to server and providing common functions to player features. The core was kept backwards compatible. Old players could be updated with a new core to, fix bugs and add new features to them. Significant difference from HTML to using Flash was testing the work only once and being confident it will work. GFP was used as a base for roughly thousand e-learning courses [11].

Development of the third version was started, but stopped in 2013 to focus all efforts to GHP. Many of the standardized features of GFP are implemented and developed also to GHP. The next subsection describes the conflicts in changing the devkit technology.

2.5 HTML5 vs. Flash

This subsection explains the difficult turning point of HTML5 taking over Flash as the e-learning technology from Prewise's point of view. After smart phones got popular, there was a dilemma for years when Flash did not work in mobile environment, but corporate users had old browser versions [11] that did not support HTML5. Therefore neither of the competing technologies worked in both worlds. It was only a few years ago that Flash was still the de facto technology for creating animation and very typical to build entire client side with it [5]. In 2009 13 out of 15 top e-learning development authoring tools outputted Flash format [12]. By 2013 the balance had shifted. 75 % of the answerers in eLearning Guild's survey considered publishing to mobile important against Flash's 62 % [13,38]. The difference is surprisingly small, but the mobile use was still less than 20% in Finland back then, in 2013, as seen in Figure 1 [10].

HttpArchive's statistics tell that in March 2015 still 33 percent of the Alexa Top 1 000 Sites use Flash [14]. This is surprisingly high considering the current rate of devices not supporting the technology. It was only a month ago I implement my latest Flash course on a client's request.

The pleasant thing about turning from Flash to HTML5 is that their similar script languages, ActionScript and JavaScript are almost the same, sharing the same programming language specification [15,178].

2.6 Gimlet HTML5 Player

GHP was already covered partly in the beginning of this chapter following the history leading to its development.

The reason Prewise developed its own player devkit was to minimize the design and development work put to every e-learning course made. The courses usually have most of the features and layout in common. So the common parts were standardized as the devkit. Standardization has been for every new feature when it was used or planned to be used more than twice. The devkit also provides functions and practices for easy customization and feature development.

As default, at least the colours of the player need to be changed to meet the customer's brand guidelines. In advanced projects all functionality and look of the player can be modified to suit client needs and requirements.

2.7 Activities

It is common to include exercises, tests and other interactive elements to e-learning courses. As Clark and Mayer (2011) state, for learning to occur, the lesson must include techniques that prompt high psychological engagement in the absence of behavioural activity [16]. But not all interactivity has a cognitive purpose. The purpose of an interactive element can also be something entirely different, like to improve usability. In a simple, but effective example, there is a hover effect in a button changing its colour and place, so that user knows his or hers gesture was detected by the system [17].

In this thesis activities will refer to engaging interactive elements or interactions added to an e-learning course to support the learning process. Activity is not a term that we use in Prewise, but felt the term activity should be used in the thesis, instead of interaction. This should help distinguish between the activities described further, and other more commonly known broader meaning of the interaction. Interaction typically means any action occurring as objects have an effect on each other. The term activity is also used in Moodle LMS to refer to same kinds of interactive elements as in the thesis. I have also found it used with similar meaning in scientific studies [18]. I will also have the term activities to include exer-

cises used in e-learning such as multiple choice tasks. The exercises are included to have better known examples of activities. However the exercise type activities are not part of this thesis.

Activities are typically an element or a set of elements that combine a functional whole. For example an activity could not refer to a single checkbox, but could refer to a multiple choice task including checkboxes, buttons and possibly other elements too. In activities the function and the graphics are standardized – though they can be customized to the client's needs and wishes. Custom activities are made from scratch and can have more illustrative graphics and roles. They are however out of the scope of the thesis.

Activities such as Match Pairs exercise engage learners more than the ones for example just showing additional information when clicked. This is because exercises require more thinking from the user. Activities can also be used to fit additional information on a page. In these cases there are usually elements showing and hiding the additional information after user activates it with a mouse action or a gesture. So far GHP has 4 standardized activities in various types such as accordion and tabs. This thesis tries to provide the best tools and technologies to develop two new activities of different types and with different interaction levels.

2.8 Animations

This sub chapter discusses why animations are used in e-learning, what types of animations there are and how they are used in GHP. Animation has here a broad meaning and refers to any animated elements in a course not just animations in the content, with an educational purpose. In general, animation can be defined as changing a graphic display.

There are several reasons to use animations in e-learning, but in the Business sector it is often just to create a bit of a wow effect and make visuals more engaging. Contrary to common belief, studies conclude that there is no understanding proving animations, in general, more effective than a series of static images conveying the same information [19]. Traversky et al. think this is because too much happens too fast [20,266]. Studies do show that animations are more effective teaching hands on procedures and tasks [19]. Also learning is more effective when narration is used aside with animation and vice versa, according to studies by Mayers and others [21].

I asked my colleagues why animations are used in courses by Prewrite. The key reasons from concept designer Hanna Nissinen's answer are to:

- clarify or underline
 - causality relations between events
 - changes of all kinds
- help understanding large entities (like animating a large diagram in phases)
- bring added value to user experience
- to raise importance of a content element by drawing attention to it [22]

Animations can be divided into two groups by the presentation format. These terms are my own, but the difference, in the context of the study, is essential. The groups are passive and dynamic. Passive animations are videos and image sequences. In dynamic animations animated elements are actively moved and modified by commands from code in real time. By animating with code one has an option for interactivity. When animations do not have interactive features or they do not need any dynamic content, e.g. localized texts, they can be created as passive animations such as videos.

Common technologies showing dynamic animations are HTML5 animation technologies and Adobe Flash. Animation technologies are covered in Subsection 2.20, General HTML5 Supported Animation Technologies.

The grouping by presentation format is not the method to categorize animations in this thesis. We are interested in the technologies used in the production the animations. I could not find nor come up with a definitive system to categorize animations that would work in this research. Because I had to use something, I decided to divide them into two main categories, informative and decorative. I also have a number of sub categories that have distinctive features. The categories are not absolute and they do overlap occasionally. Next there are descriptions for these categories. Chapter 4.4, Animations, studies the animation types selected for the thesis from the tool requirement perspective.

2.8.1 Informative

In the animations in this main category the movement or transformation has an informative purpose – and therefore not only the graphic, but the animation itself, has information that is piece of content of the course. The animation itself adds pedagogically relevant information

to the graphics. In my experience informative animations are on average longer than decorative animations and need more work. It is advised to present longer animations in steps and let learner control proceeding to the next step [23; 24]. This is to enable learning at a preferred pace [23; 24]. This feature requires interactivity. Next the informative animation types are described.

Process

Process animations are transformational visuals and tell how something works. They illustrate temporal or spatial change [25]. This category includes a wide range of different types in it. They can be for example flow charts; illustrations of mechanical, business, or scientific processes [25].

Procedure

This is similar to process because this type also includes a sequence of steps. The main difference is that procedure is directive in nature and demonstrates how to do something. As stated earlier, practical procedures are effectively taught by animations. Screen Captures are in this category. Also character animations can be used in this category.

Screen Capture

Screen capture animations are common in teaching or demonstrating use of computer and Web applications. E-learning is suitable for teaching how to use software [5]. There are specific tools for producing this type of animations.

Character Animation

Character animations are typically cartoon like human, animal or fantasy characters. Although using characters is an efficient way to apply the Personalization Principle in e-learning [26], it is not frequently used. In my experience there are two common reasons not to use animated characters. Sometimes they are just not needed, and second, using them requires extra work. So they might not fit the budget. There are also other reasons to use them, for example to illustrate a procedure or phenomenon or they can be just decorative or engaging. When characters are used for illustrative animations they are visually more realistic. In Prewrite's courses characters are often cartoon like.

When the character has a tutoring role they are called Pedagogical Agents and they guide the learning process. They then have a speech audio or text. Using audio with a character improves learning. According to studies characters life-like look is not essential for learning, but they should move in a human-like behaviour. [27]

In GHP the characters often have a role being Pedagogical Agents. Character animations are often somewhat interactive, meaning they tend to react to events the user creates. This is of course very dependent on what is the role of the character in the course.

2.8.2 Decorative

This is the other main category of animation. The movement and transformation does not have informative purpose in this category. Therefore the purpose of animation is mainly decorative – in other words, to please the eye. The graphic itself can be informative though. In my experience decorative animations are shorter than decorative, on average. As an exception intro animations used in the beginning of a course can be very long. But they can also have a partly informative function. The most common case of decorative animations is a graphic animated to appear on a page.

Charts and other diagrams

The definition of word chart is not clear and it overlaps with terms diagram, graph, plot and even map. But by chart, I loosely mean structured information represented in graphical form. I concentrate on the types of charts that can be easily rendered from data, such as pie charts, histogram or density maps. Charts are informative, but animating them is mainly decorative.

A chart graphic can consist of many elements so there can be a lot of work animating it without any assisting tools. Based on my research there is plenty of tools to animate certain types of charts.

Wow –animations

This category is not based on any classifications found in literature. I came up with it, but when it comes to animation tools and techniques, I feel the category has a *raison d'être*.

Animations in the category should look especially good – to make a wow-effect. To do this one might need some special tools designed to create some specific very cool looking effect. Introductory animations often fall into this category, because their purpose is usually to wake up the learner, engage and make to the topic seem interesting. 3D usually creates a wow effect, but using it requires lots of work. This category requires often both very good graphic design and advanced technical implementation.

2.9 Client-side Requirements

The technology requirements of the client devices for GHP also apply when selecting the tools and technologies in development, because they should meet the same requirements.

Although HTML5 is nowadays widely supported – especially compared to Flash – the reality is that the support is not wide enough in the old computer browsers [28]. HTML5 standard is still evolving and none of the current browsers support HTML5 standard fully at the time of writing this [28]. Supporting wide range of devices and systems takes a lot of work getting the compatibility issues solved and tested.

The goal in the web in general is to support all devices with internet connection and a browser [9,36-37]. Against common practice GHP developers decided the support more restricted range of systems and devices. The main reason to do this was to assure proper quality and user experience to the clients, also to save time in development and testing. Restricted support has not been an issue so far.

As default the course content and layout is optimized for devices with larger display than smart phones, meaning tablets and computers. Courses should be viewable by smart phones though.

Resolution is a major cause of problems in mobile [9,51]. To limit the problems, GHP has two minimum limits for supported resolution. As default player is tested and guaranteed to work in resolution 980 x 550 pixels and larger. With smaller resolutions down to 360 x 480 pixels, the important content will show, but will not be optimized unless otherwise agreed with the client. GHP does not have any tested support for special devices such as browsers in cars or TVs, smart watches, Google class etc.

GHP client-side requirements are listed in Appendix 1, GHP client requirements

2.10 HTML5

HTML5 is of course the main technology which made it altogether possible to develop a full featured e-learning player supported by Prewise's clients' devices and environments. HTML5's origin is in W3C and the earlier versions of HTML. W3C released its first public draft of the standard in 2008, and it took years to create and publish the final recommendation in 2014 [29].

HTML5 is often considered also an umbrella term for the next generation web standards including CSS3, SVG, Web SQL, WebGL, and Woff among many others. These unofficial sub standards are not all final recommendations like HTML5. Of these, only SVG and partly CSS3 are subjects of this study. W3C is already working on HTML5.1, so HTML is not ready at all. The version numbers of HTML are not important in development. Developer can use features as they are supported in browsers. In general HTML5 is well supported in mobile devices and all modern browsers. I suppose HTML5 support defines a modern browser.

The key HTML and relative standards and elements that made developing GHP possible are:

- Video element
- CSS media queries and responsive web design?
- Web font support

Also jQuery and GSAP play a big part in developing light weight interactive standard and custom activities.

2.11 Best Practices

A few very basic level best practices for Web development are:

- Keep it simple.
- Reduce the HTTP requests to the minimum.
- Make the cache a friend. [9,689]

These guidelines are followed in developing GHP, but sometimes it is difficult or even impossible to keep things simple in a devkit that should have an extensive set of features to support variety of different content and concept designs. Currently the devkit loads code for

all features it support. As there are more and more optional features added to the player, there is a plan for a functionality that would support loading the code for only the features that are used in the course in question. This has not been implemented yet, but would help apply two of the suggested best practices. The most important design pattern used in GHP is Responsive Web Design. A number of other best practices are mentioned throughout the study.

2.12 Responsive Web Design (RWD)

It is said that Responsive Web is a response to requirements from the mobile use. The mobile Web use introduced small screens to modern Web and made the resolution spectrum broad in the smaller end. In my opinion the displays nowadays vary so much in size that even without the mobile devices, there would have been a need to have the layouts adjust to different displays and environments.

Elements on a responsive page need to adapt to the constraints set by the environment.

The constraints are:

- Display dimensions
- Display quality (pixel density, colour capability)
- Connectivity (network conditions)
- Input types (touch, mouse, keyboard) [30]

By optimizing the page to work best in the conditions set by the constraints, it provides the best clarity of content, usability, loading time, and performance [30].

Implementing a Responsive Web Design (RWD) to a site allows it to adapt to viewport sizes of all sizes. Typically the site should do this without redirecting mobile users to a different URL. Having one site for all users, maintains the same look and feel for all users [31,8]. Despite all the commotion around RWD, it is not suited to every situation. In some situations it is better to assess carefully whether RWD is the better choice over a separate mobile site. With poor design and/or development an RWD can cause issues in a high performance site [9,136]. There will be more about this subject later in this chapter under the title Mobile First.

Two essential concepts that affect how a web page is rendered need to be understood to understand the issues that RWD helps solving: canvas and viewport. In this context canvas

does not refer to the HTML5 canvas element, but to the “surface” that the page is rendered on. The canvas has an infinite size, but only visible part is what is covered by the web page.

The viewport is harder. W3C defines the viewport as “a window or other viewing area on the screen through which users consult a document” [32]. In the framework of this study it is an area on a screen that shows the user a web page or a part of it. If only a part of a page is visible, there is usually a scrollbar available to access the rest of the page.

For a long time, until 2007, in browsers the viewport size was always the same size in pixels as the window size or the screen size in typical mobile devices. Then came Safari for iPhone and changed it. The mobile browsers started using a viewport size bigger than the screen resolution. In the case of iPhone 3GS the viewport width is 980 pixels which is scaled down into a screen 320 pixels wide [9,200; 33]. Lately the scaling has been to the opposite direction in the large high density screens in the smart phones. An example of this is the iPhone 6 Plus with a viewport 414 pixels wide scaled up to the screen with 1080 pixels width [33; 34]. Because of the high pixel density the text and other elements would be too small to read properly without scaling them up.

There is an HTML Meta element for defining settings for the browser on page’s dimensions and scaling [35]. In RWD it is preferred to work with the real pixels instead of the virtual “zoomed” pixels. This can be achieved with ‘width=device-width’ setting in viewport tag. With the setting there is no default zoom factor used when the page is rendered and shown on the screen and RWD can be used to optimize the content to the screen.

The key components that make RWD possible are the viewport tag and CSS3 media queries [36,8]. The CSS3 media queries enable setting CSS rules that apply only in certain viewport sizes or size ranges [36,8]. This allows using different CSS rules in different viewport or screen sizes. Bigger text size is definitely required in high density mobile screens, than in older low density desktop displays.

One can use a RWD framework to minimize time spent inventing the wheel again when developing a RWD project. Benefits of using a RWD framework are:

- Saving time
- Support of a community and extensions
- Cross browser compatibility
- Documentation [31,15-16]

In general the frameworks suite better for web sites, but we use one in GHP too, Bootstrap. Thing is though, we do not actually use it for RWD, but only for styling 'select' elements. Next the key concepts and aspects of RWD are described in more detail.

2.12.1 Responsive Layouts

A common design pattern included in RWD is using a fluid (or flexible) grid layout. This means the page elements are placed to an even sized invisible grid that uses percentages for widths. Basically this grid is a pattern of columns. Each element in it is set to take a width of a number of columns. [37] The total column count on the page can be anything, but 12 seem to be the most popular. The elements next to each other in a 1200 pixel wide layout would be too narrow in a layout fit to a 480 pixel wide display. With CSS media queries layout styles can be bind to breakpoints. In RWD breakpoints are normally certain viewport widths and/or heights. When a breakpoint is reached, elements can be rearranged with the CSS styles active on that breakpoint. Changes in a browser's width can activate breakpoints rearranging and modifying the elements on the page as shown in Figure 2.

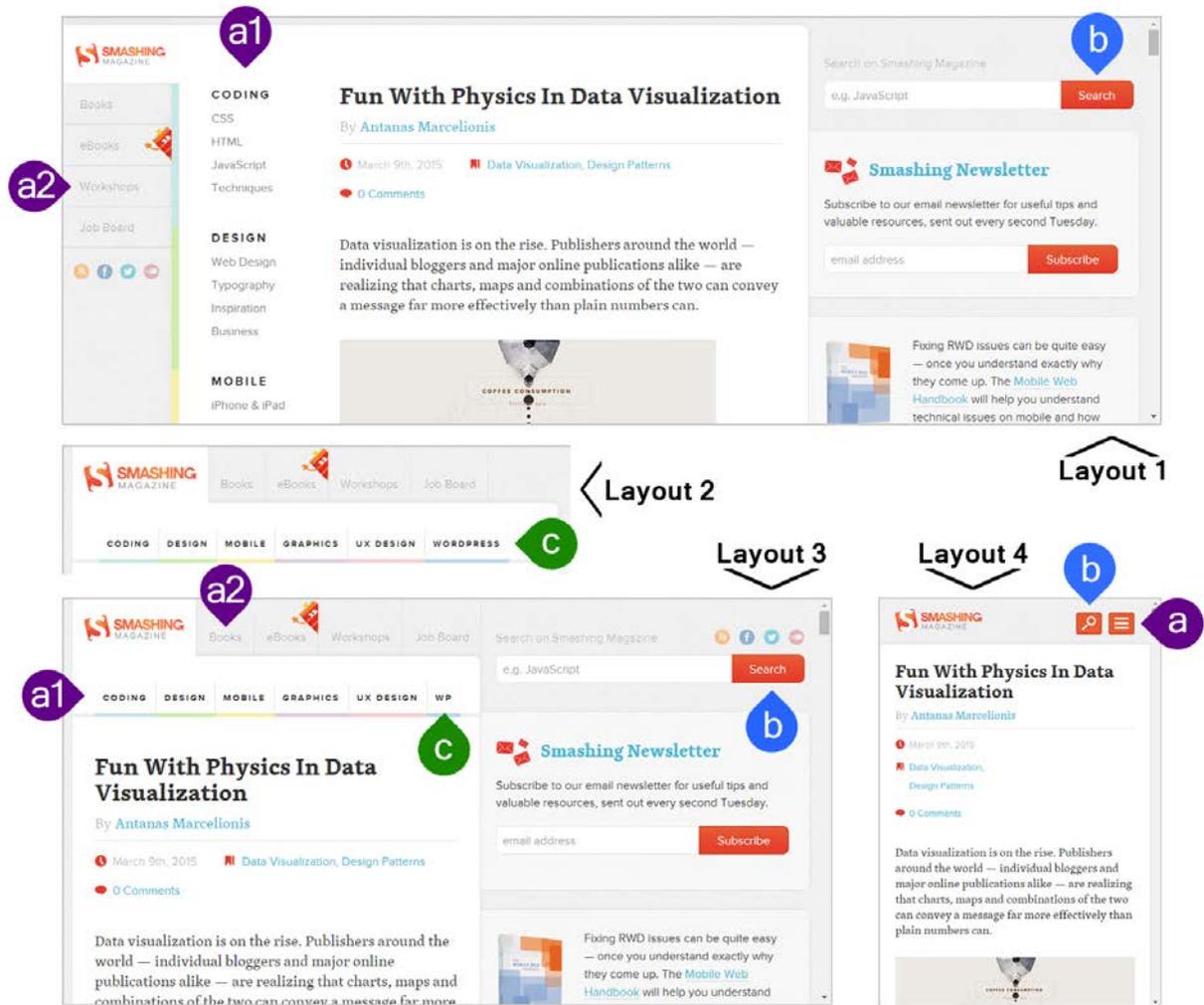


Figure 2. Four examples of a responsive UI in different size viewports showing examples how the UI can adapt to the space available. Screen captures are from <http://www.smashingmagazine.com/>.

Examples of how a responsive layout can adapt to different viewport sizes are given in Figure 2. The menu elements a1 and a2 are placed on the side of the main content column in the layout version 1. They can be placed there because there is enough space available on the side. This way they free the most space for the main content in vertical direction. This minimizes the scroll and therefore increases usability. In layouts 2 and 3 menu elements switch from vertical to horizontal shape and move above the main content column. Also the subsection titles shown in menu a1 in layout 1 are hidden in smaller layouts. Hiding less important elements such as sub menus and “mood” pictures is one of the principles of Responsive Web Design. Interesting detail in layouts 2 and 3 is also that the label “WordPress”, in menu items (c), is shortened to form “WP”. This is to fit the menu to the narrower viewport as a responsive feature. Layout 4 emulates a mobile device. In it, the menus are hidden and shown after tapping the menu icon (a). The site’s search feature (b) changes

from the visible form elements in layouts 1-3, to an icon in layout 4. As part of the RWD best practices, the buttons are often shown larger for mobile devices. This is to make it easier to hit them when tapped with a finger on a small screen [31,169; 38]. Use of this practise is not seen in the Figure 2 though.

GHP does not use fluid grid. It does have a 2-column system for its content area, but the width ratio of the columns is not fixed and can be set in admin side. Therefore it is not comparable to a common grid system. Otherwise GHP layout is very responsive and it has 5 active breakpoints changing the appearance of the layout.

2.12.2 Mobile First

Bases to the subject of this chapter come on the topics discussed in Chapter 2.3, Mobile Web. The data shown in the graphs in Figure 1 visualize how desktop, mobile, and laptop shares from all internet usage have developed in Finland and Europe. The mobile use will soon overtake desktop use. This is one of the reasons why the “Mobile First” design practice is growing popular. GHP was not developed with mobile first approach, because most of the users on the courses still use laptops or desktops. So far this has not been a problem.

The idea and the term “Mobile First” was invented by Luke Wroblewski. The idea in simple is to create web projects first for mobile devices. This has not been left to be a just an idea, because companies such as Google, Facebook, and Adobe are already applying this approach in their products. This approach tries to tackle the problems born when developing “desktop first” and then mobilizing or even just “minimizing” it for mobile. Desktop first can lead to a poorly optimized desktop site and a terrible mobile site. [9,140]

Firtman (2010) claims, “Some statistics even indicate that users tend to choose desktop web versions over mobile versions when using smartphones.” [9,36]. If this is true, I believe it is due to bad implementation of RWD. Probably hiding import content or features from mobile users. For user, it is important to get the full experience also with mobile device. This requires well implemented RWD.

Firtman (2010) suggests that it might not be the best idea to have the same site adjusted to both desktop and mobile using RWD [9,137]. I have to disagree with him on this. As

Wroblewski (2012) clearly illustrates in his Display size continuum in Figure 3, there is no clear distinction in screen sizes between the “desktop/laptop” and “mobile” devices [39,81].

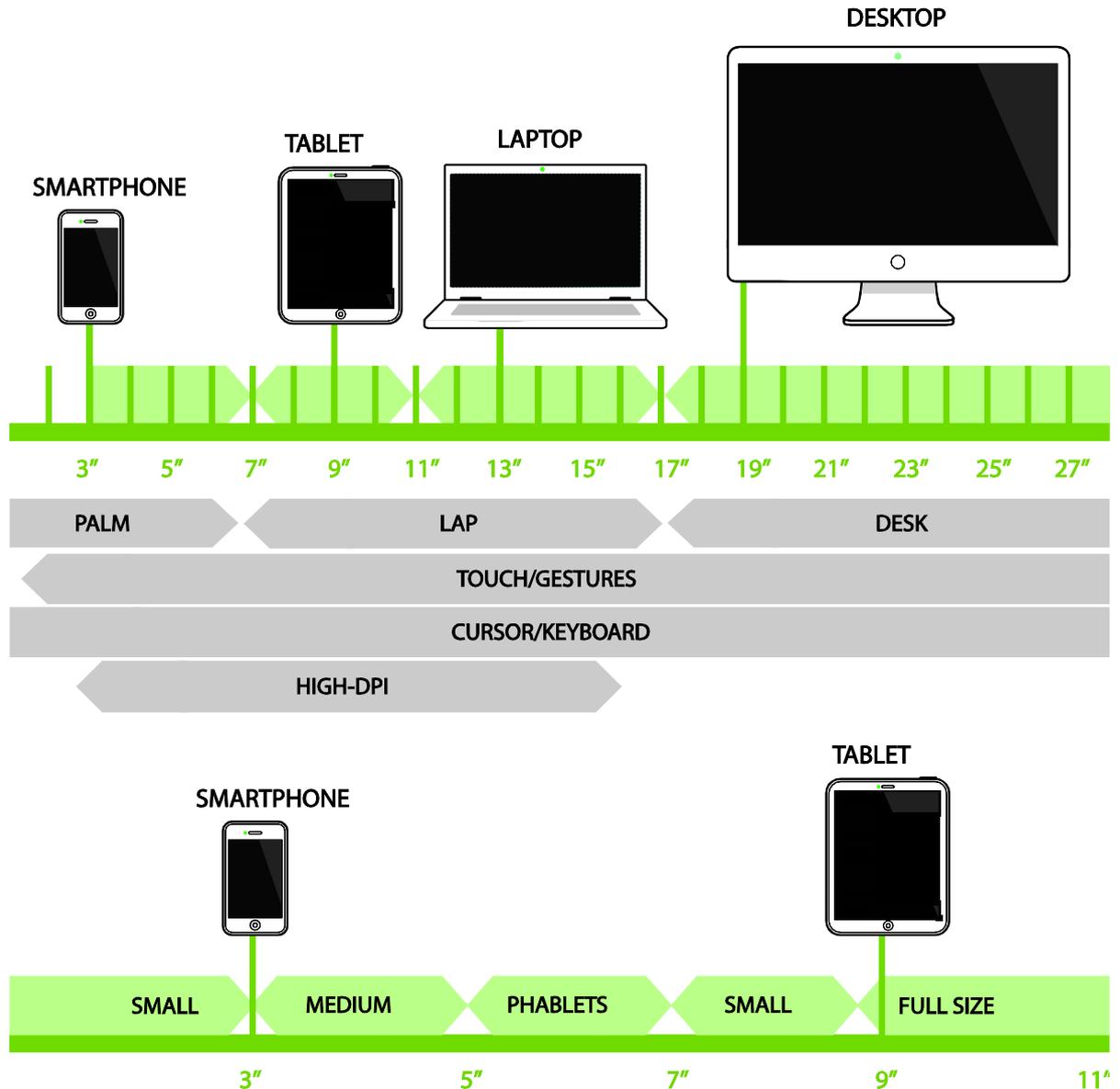


Figure 3. Display size continuum and similarities and differences of the mobile devices and computers. Modified from Wroblewski (2012) [39,40; 39,43; 39,81]

Upper part of the Figure 3 illustrates how display sizes continue where the smartphone sizes end. The laptop sizes continue where tablets end and so on. The lower part of the figure shows in more detail how smartphones grow seamlessly to phablets and phablets to tablets. I would even go further than the figure suggests and claim that in many cases the size ranges even overlap. Large tablets can have larger displays than small laptops.

I see the size continuum as proof that there is no reason to make two sites. If it is not possible to clearly identify to which group the device belongs to, why not just have them all in the same group. I suppose if there would be a clear distinction between groups, there would also be some reason to have separate sites.

To provide the above mentioned full or best experience possible, the developer could use a technique called Progressive enhancement. It means taking advantage of the client devices and browsers features that might enhance the experience for the user. All basic content and functionality would be available to every user, but there would be also an extra layer of advanced layout and behaviour for those who have the support for it. [9,132] An example of this can be showing the user on the map, if device has a GPS and browser supports it.

2.12.3 Responsive Images

Like other elements on a responsive page, also images need to adapt to the constraints set by the environment. Responsive images do that by use a group of techniques. These are resizing the images on the server-side, CSS scaling the image in the browser, and masking the image with CSS.

It is of course possible to change the size of the image in the viewport, but it is always the same image file loaded to the browser. This means one needs to use the image – even in the small phones – in the largest resolution estimated any of the devices will require. This is waste of bandwidth. W3C has proposed a new element called <picture> to patch this flaw, but it is currently supported only in Chrome and Opera and latest Firefox, MS Edge, and Safari. It can be used with a fall back to HTML image element. [31; 40]

Server-side resizing is for not wasting bandwidth with slow connections and providing images that match to the requirements of the screen size and pixel density [9,138]. This has been implemented to neither GHP nor Gimlet. My feeling is that now would be a good time to implement support for it as latest versions of all major browsers started supporting it. There is also a methodology for making an assumption of the bandwidth from the resolution of the viewport and matching the image size to the bandwidth speed. We are not using the method because we feel it does not work anymore. Today the variety of screen sizes, viewport resolutions and connection speeds make such a mix that it is better not to make any assumptions. Assumptions lead to failure too often.

What we are using in GHP from the responsive image tool palette are the client-side techniques. We always get the image in the same size from the Content Management System (CMS) image bank. So far we have used images saved from the layout for the large viewports. This means images are not scaled for the big displays, but need to be scaled down for the small ones with CSS. However, the small displays with high pixel density can benefit from the large image, and show it really sharp despite the relatively small viewport resolution. Also there is an option to use the masking method for page's main images. See Figure 4 for examples on how the image proportions are not fixed, so its shape, along with the size, can adapt to different viewport sizes with the layout in this method.

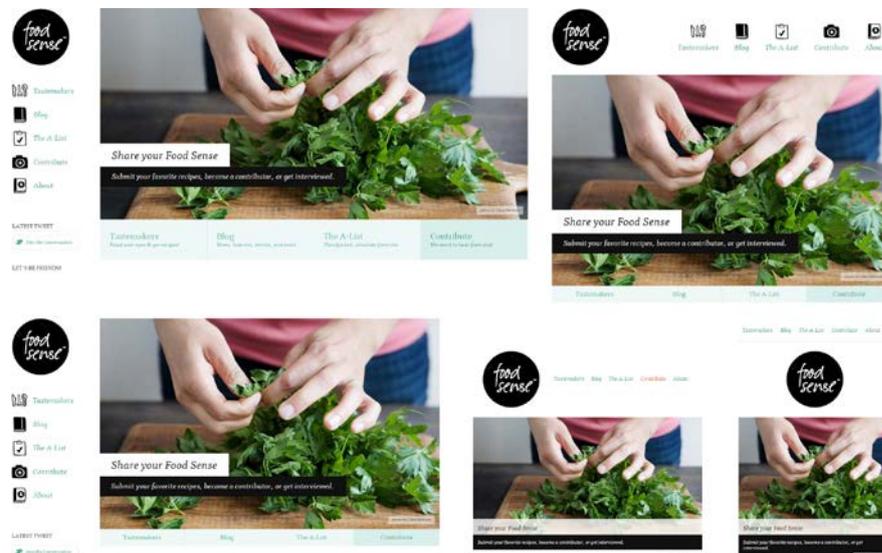


Figure 4. Five examples of the Food Sense website in different sizes demonstrate how a responsive image adapts to the space available using masking and centering [41].

The use of the cropping effect requires cropping the original image. This requires a “margin” area on all sides with non-critical information that can be cropped off in the browser.

2.12.4 Problems

RWD does not come without issues and problems. When having separate sites for desktop and mobile, the hacks made for old desktop browsers do not need to be loaded for mobile browsers. The desktop users need large images for good quality in their up to 28 inch 4K monitors. However there is no need, nor sense, loading images that big to a mobile device with a small screen. Images are not an issue when responsive images with server-side resizing have been implemented to the site. If some less important content is hidden on

small screen sizes on RWD site, it is waste of bandwidth to load them in the first place. [9,137]

In my opinion the most important difference between desktop and mobile causing problems for a one-for-all RWD site is not any of the static device properties, but the bandwidth of the connection. A site used by mobile users should be built this always in mind.

2.12.5 Future

RWD is still evolving as the standards and browsers develop enabling better responsive frameworks and implementations. I came up with useful future responsive feature, where a site could read the devices accelerometer and interpret from the data whether the device is shaking or bouncing. Then increase the text size the more the device is shaking for better readability when walking or on a bumpy road.

Setting width as percentages was mentioned with fluid grids. With percentages we can set relative proportions. This leads to elements scaling with the element that they have the relation in size. Making elements scale by their parent element or viewport size is key techniques of RWD. With bitmap images scaling causes all sorts of problems as seen earlier, but with vector images most of the problems do not exist. A further Subsection 2.18, SVG, is about using vector graphics in the Web with SVG. With SVG and some CSS it is possible to create an image that changes the graphics in a responsive way. This way it is possible to hide elements or details as it is shown on smaller displays. Joe Harrison has studied this in his project Responsive Icons [42]. I, with many others, believe use of SVG will grow significantly in the future [43,xiv-xviii].

2.13 SASS

Syntactically Awesome Stylesheets (SASS) is a CSS metalanguage and preprocessor. It is interpreted or compiled to standard CSS. SASS works on Ruby. It was selected to be used because our main developer prefers it over LESS, had more experience with it, and It also comes with Yeoman which was selected as a workflow [44]. GHP utilizes SASS features extensively and it has proofed to be a very practical and useful tool in the development. It helps a lot in keeping the CSS code shorter, cleaner, and better structured for the developers.

The most useful features in SASS are variables, mixins, nesting, and operators. Variables are useful in the same way as in any programming language. GHP benefits from this a lot. The default values for all variables, such as colours or margins, are defined in the core, but they can be overridden in client specific customized players. By using variables, values can be changed to a number of styles from a single line. Mixins enable setting functions for repetition. Mostly they are used to add vendor prefixes, but they can be used for more too. Compass is a plugin used with SASS with plenty of useful mixins.

Nesting helps structuring the selectors following the same hierarchy as HTML they apply to. It also decreases the amount of code in the source files. Operators are especially useful with variables. GHP uses operators for example with margins. There is a variable having a pixel width for layouts marginal "unit". In the styles the unit variable is used with a factor. A value for a margin can be N times the base unit. If smaller margins are needed in the customization only the value of the margin unit can be changed.

2.14 MVVM and Knockout

MVVM (Model View ViewModel) is an architectural pattern that derives from Model View Controller (VMC) pattern. It was named and defined by Microsoft to be used in Windows Presentation Foundation (WPF) and Silverlight development. MVVM was soon adapted by Adobe Flex developers. Now the pattern has been implemented as a framework for JavaScript in libraries such as KnockoutJS, Kendo MVVM and Knockback.js. [45,92-94]

MVVM is used to separate the UI (the View) from the business logic (VM). The View formats the data for user. It is the layer that is visible for the user and it interacts with. [45,92-94]

The Model represents the data or information of a "business object" – a course in GHP's case. The ViewModel (VM) is there to connect the View and the Model by providing the right selected data from the Model converted to a format and structure suited for the View [45,96]. The idea of connecting the properties and events on the HTML elements to the methods and data of the VM is a called binding or data-binding [46,5]. In GHP the main VM holds the logic and data of the active page in the course.

Benefits of MVVM are:

- It provides means for developing the UI (View) and the code with its logic (View-Model) in parallel.
- It makes the View abstract and so makes it easier to manage without the business logic.
- IT enables easier unit testing for business model than in event-driven code. [45,98-99]

There are of course disadvantages in using MVVM. It does not suit all situations. It is over-kill when used in a simple UI. The data-bindings can be harder to debug than imperative code. Using MVVM can require a lot of book-keeping code in data-bindings. Designing the ViewModel for a large application may require more effort in advance for the generalization. [45,98-99]

MVVM pattern is implemented to GHP with KnockoutJS, later just Knockout. It is a free open source JS framework making the UI dynamically communicate with the data model. Using Knockout makes it easier to remain the code manageable, stable, and maintainable.

Three core concepts of Knockout are partly the same as MVVM's:

- Dependency Tracking with Observables
- Declarative Bindings
- Template Support [47,13-14]

When Knockout is used, the View, in MVVM, is the HTML and the bindings linking it to the ViewModel. The ViewModel is a JS module providing methods and dynamic properties for the View. The library is actively developed and well documented with good examples [48].

The framework works best in single page applications like GHP. The API can be hard to scale so there can be better options when developing big applications. Also there has been criticism against the lack of guidelines or best practices in the application structure [49]. I must agree with these critics. Knockout does not provide any guidelines driving to develop a view model structure that will be maintainable. Therefore it is possible for an unexperienced developer to end up with large unmaintainable functions.

The reason why MVVM and Knockout were selected to GHP partly originates to Prewrite's long history of using Microsoft technologies in server-side. Knockout was developed by a

Microsoft employee Steve Sanderson. Gimlet's main developer, Martynas Danilevičius, suggested choosing MVVM and Knockout. The other reason was the popularity of knockout. [44]

In the thesis KnockoutJS is used in the activity templates and the required bindings. Each new standardized activity type also has its own ViewModel.

2.15 Asynchronous Module Definition – AMD

AMD is a specification to implement module pattern design pattern in browser with JavaScript. With AMD developer can define modules and dependencies that can be loaded asynchronously [50]. Implementing AMD improves performance by having modules in separate files and loading them at the same time and only when they are required instead of loading them one by one as shown in Figure 5 [51] [52]. AMD prevents polluting the global namespace by encapsulation of the module [52] and makes debugging easier [53] [54].

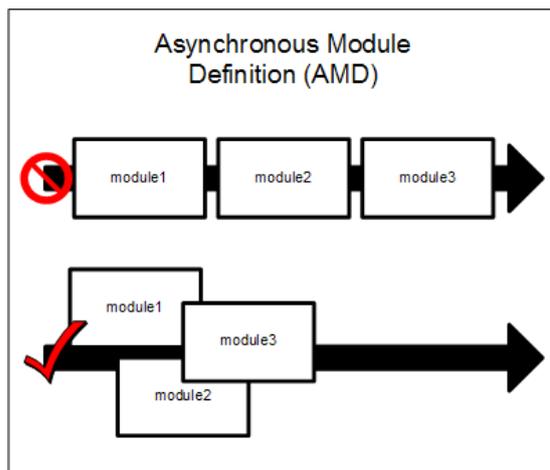


Figure 5. Differences between synchronous and asynchronous loading [51].

AMD is implemented in GHP with RequireJS. This means AMD modules are managed and loaded to the player using RequireJS. It is one of or the most popular dependency managing frameworks for browsers [55] [56]. That was the main reason for choosing it to GHP [44]. The new components resulting from activity tasks of the thesis are also added to GHP as AMD modules.

2.16 Other Tools

Many of the tools presented in this subsection were selected by the projects Lithuanian main developer, Simon Čereška, and former head of development, Eetu Tuomala. The choices were made before I got involved with the project. I interviewed them for the reasons why they ended up this set of tools. I have understood that in general we have tried to use popular tools that are somewhat best practices in web development or masses have already found useful.

The project uses Git to store, version and distribute the sources. It was selected for its popularity. WebStorm is the main IDE for the programming for the developers. Npm is used for package management in the development environment. Yeoman workflow is used in the development. It was selected to GHP because it seemed easy to get started with [44]. Yeoman includes three types of tools. Bower manages the assets and components for the project, such as jQuery and RequireJS. We switched from original Grunt to more effective Gulp to run tasks, such as minimize, publish, and preview.

The project uses Jenkins as the continuous integration tool. Jenkins runs for example automated tasks when a new release is made. The plan is to run the future unit tests on Jenkins. It was selected for GHP development after an evaluation also including Gitlab CI and Atlassian Bamboo early 2015. Gitlab CI seemed too confusing based in the documentation and Atlassian Bamboo provided nothing special as a paid tool compared to Jenkins. Travis, TeamCity and alternatives were not an option as we needed the CI installed in our local network accessing our local Git-service. [57]

The main client side JS library is the obvious jQuery. JQuery is the most popular of all JavaScript libraries [58,216]. This is probably because it provides a concise, extensive, and flexible tool to access and manipulate DOM elements. The general info about it will be skipped assuming it is common knowledge. JQuery overcomes cross-browser compatibility issues increasing productivity. Its selector engine finds DOM elements by setting CSS-style selector statements. [58,19]

Using jQuery does not come without drawbacks – a major one being performance. Some of jQuery features are faster done with native DOM methods and properties [58,22]. Also in some mobile devices the parsing of the jQuery framework can use 0.01% of the device's full battery [9,455].

JQuery is used extensively in GHP. Also a number of its plugins are utilized. JQuery will certainly be benefited in implementing the activities.

Modernizr detects and provides information on supported browser features. It was selected to GHP because it is part of HTML5 Boilerplate and can be considered as best practice [44]. Hammer.js is a library for gesture recognition. Hammer got selected to GHP after good experiences with it in earlier projects and because it is actively developed [59]. GSAP is a popular and very extensive animating library. It is covered more in detail in Subsection 2.20, General HTML5 Supported Animation Technologies.

Video.JS is used in GHP as the video player. It was selected for its popularity at the time and it met the requirements [44]. Video.JS has a lot of useful plugins that extend the basic functionality. For example one plugin enables linking YouTube videos just by their common page URL. VideoJS Cuepoints plugin was used in GHP's video player. It enables synchronizing actions with the media timeline.

2.17 Testing

Testing is very poor still at this point, but we have plans to set up automated unit tests that would run before each release. This is a required professional approach to proper testing. We also have two test courses that new builds can be tested against. One course includes examples of all our standard features, and the other has all sorts of custom functionalities that take advantage of GHP's functionalities for customized features and content. We also plan to start using a checklist making sure all required test have been executed and successful, to ensure high quality.

2.18 SVG

Standard Vector Graphics (SVG) is W3C standard for 2D vector graphics. It is very old for a Web technology, dating back to 1999. SVG is getting now a second coming with HTML5 and I believe its role in web graphics is going increase even more in the future. Explosion of the active use might keep us waiting until the support for the future 2.0 specification though.

SVG suits especially well to the requirements of the responsive web due to its vector based nature. It scales to any size without losing any sharpness. The vector format makes it also very lightweight especially when considered it can be scaled to any size keeping sharpness. There are tools to optimize the file to even smaller file size. The markup is XML-based and therefore is easily generatable. SVG it can be styled and modified dynamically through its DOM with CSS and JS. SVG is easy to animate [60] making it a promising candidate as technology used in the animation tasks of the study.

The study focuses on SVG animation tools. This is because based on the information and experiences working with it; it suites the best for modern Web. The first reason is that it is very light weight compared to other technologies due to its vector based nature. Therefore it suits efficiently on mobile use. It still supports bitmap graphics when required. The second reason is that it is truly dynamic graphic format when used with a manipulation library. It can have dynamic content and it can be manipulated in real time. The third reason is also linked to vector format and it is the scalability. Nothing scales as well as vectors keeping the sharpness of the graphics. SVGs can even be used responsively so that the graphics change depending on the viewport size or other properties. Details of the graphics can be hidden for smaller displays with CSS.

The weakness of the SVG use is the animation editors, they are not very efficient in general. Therefore animating SVG often requires coding skills. The browsers also do not seem to have full support for all features of SVG. Often an alternative way is needed when trying to implement something using an advanced feature of SVG. This is because the easiest or the most suitable way may not be supported by some of the modern browsers [61]. Especially IE, but also Edge has work to do with its SVG support. None of the browsers support SVG specification fully.

2.19 SVG Tools

Adobe has an extensive collection of applications that can be used to create and implement SVG images on Web. Illustrator is the industry standard for 2D vector graphics. Adobe Edge Animate and Adobe Animate (formerly Adobe Flash Professional) can be used to develop Web sites with SVG. I must still state that after preliminary tests, their support for manipulating SVG is still unsatisfactory. In Edge Animate manipulating a specific SVG element is not supported – only whole SVG images. In Animate it is possible, but only with a plugin. In preliminary tests, I run into other obstacles for using Edge projects in our envi-

ronment, as it exports its assets as separate files. If assets would be added to our CMS media bank, the references to them will not work anymore. Also the localization of the texts is problematic when they are embedded in the code exported by the software. The greatest benefits of using this software are having the WYSIWYG editor for the Web graphics and a timeline for animating. Development of Edge Animate has been ended.

Raphaël is a SVG and Canvas drawing library. The library has been popular due to its easy syntax [62,12]. Raphaël's great support for old browsers is growing less important though. It has been replaced by a significantly lighter Snap.svg, with only support for modern browsers. They are both developed by Dmitry Baranovskiy.

Snap.svg

Snap.svg is a library for manipulating SVG – it could be considered as the jQuery for SVG. It also has a method for animating. The motive for developing this library was getting a library that would take advantage of the better support of SVG in modern browsers.

GHP already uses Snap.svg to perform two tasks that enable us take full advantage of SVG. Animating SVG elements require it to be inline in HTML, but in GHP the SVG files are loaded from LMS media bank. Therefore they could not be animated when they are just loaded traditionally to an img element. With snap SVG we can read the file source and append it as inline HTML, enabling access to its DOM and animation. Snap.svg is also used for wrapping multi-lined dynamic texts in SVGs with a self-made plugin. This is because, as strange as it sounds, SVG does not support auto wrapping in multiline texts.

Wrapped text

I feel that the biggest problem with using SVG in e-learning is its lack of support for wrapping text to multiple lines [63]. This may not sound like a big issue. The text can actually be divided to separate lines of text manually, with tspan. But it is not that simple anymore if taken into account that courses should be made easily localizable. For easy translation the texts should be separated from the SVG code. And they should be kept in the same blocks or entities they are shown to the user. For this reason it is better to have multi-line texts auto wrapping.

SVG Optimization tools

I do not have data on what is the most popular IDE for designing SVG graphics. I would assume it is Adobe Illustrator, because it is the industry standard for 2D vector graphic design. The SVG format saved from Illustrator is not fully compatible and optimized for Web use. Illustrator uses 'mask' and 'use', elements that are not fully supported by IE. Also it occasionally adds unnecessary elements that increase the file size. It can also alter the group element ids. The file size of an SVG can be decreased with SVG optimizers, of which the SVGO and its versions are probably the most popular.

Find next a short introduction to alternative animation technologies in the Web and reasons why they do not suit modern web animations in general as well as SVG.

2.20 General HTML5 Supported Animation Technologies

When I started developing HTML5, after doing Flash for 8 years, I felt as if HTML5 was not ready for long, impressive or advanced animations. Since then I have learned that it is partly true. There are APIs, JS libraries and CSS techniques for creating moderate animations. The most advanced HTML5 animation features cannot be used, because some of the browsers are missing the support for them. One also needs to know what specific technology suits for each purpose, because there is no one tool covering all animation needs. The tools and the libraries that are used in longer informational animations should support animation controls for at least pausing, stopping and resuming.

Video

I suppose the most straight forward way to add an animation to a web page is to add it as a video. This method does not require any coding skills and it should be supported well in older browsers too. It would come with built in controls for the user to control it. Using video as animation format in modern e-learnings has its down sides. The first is scalability. To be able to provide a good quality for large displays, HD video is needed. Full HD video requires quite a lot of bandwidth which is an issue for mobile or otherwise slow connections. The second problem is it cannot have dynamic content and it cannot be modified real time. Well, I suppose both problems could be overcome with an advanced streaming server, but using one is expensive and time consuming.

CSS

CSS3 has two sub specifications CSS Animations and CSS Transitions for describing declarative animations. These would be a great technique, but it does not have support with SVGs in IE.

Canvas

Canvas could be considered an alternative for SVG. But SVG suites better for the processes we use in Prewrite, because of its XML format and better support for interactions as well. Canvas is also not particularly well applicable for responsive web. This is because it does not scale as efficiently, due to its bitmap nature. Canvas can be resized and redrawn to adjust to a responsive layout, but this requires scripting and recalculation of the graphic elements to the new size and position in canvas [64,114-115]. Canvas is not accessible by screen readers. SVG is, because its elements are part of the DOM. [65]

WebGL

WebGL is a web standard for low-level 3D graphics in the browser. It is a JavaScript API enabling hardware-accelerated interactive 3D graphics with the canvas element. It is developed by the Khronos Group. It does not suit in GHP use, because it is not supported in IE 9 and 10.

SMIL

Synchronized Multimedia Integration Language (SMIL) is a XML-based language by W3C for defining animations. It is partially included in the SVG specification. [43] This would be a very natural and easy way of doing simple SVG animations in my opinion, but unfortunately IE does not support it [40] and Chrome and Opera have set it as deprecated. It is expected that support for it will be dropped in the future. Therefore this thesis will not study it further.

The Web Animation API

This is a new web specification for an animation API by W3C. It is still a draft, but it is already supported in Chrome and Opera. It seems it is not yet well known to the community, because I only found a few mentions of it when doing the research for the study. When implemented into the browsers, the API will provide a unified language for CSS and SVG an-

imations in an animation engine. It will support synchronized animations running in their own threads and nested animations that are not currently possible in HTML5. [66; 67]

JavaScript

JavaScript is obviously also an animation technology. Not only is it required as scripting language for Canvas and WebGL, it is also very handy and popular for just animating any DOM elements by manipulating their properties. It is recommended to use a library if movement from one state to another should happen over a period of time. The library can interpolate the animated element property values for each state in defined refresh intervals. Animation libraries are certainly used for more advanced purposes too.

GSAP

GSAP is an HTML5 animation library suite that consists of several individual libraries that work together. GSAP is widely considered as an industry standard [66]. It was such already in the Flash era. The main library for basic animations is TweenLite. TweenMax extends TweenLite and comes with advanced animation features. GSAP supports synchronization of animations through its timeline libraries. TimelineLite and TimelineMax are sequencing tools that enable joining individual tweens and controlling them as one timeline with even a time seeking feature and a reverse mode. There is also 19 plugins that add specialised and even more advanced features and enable using GSAP with third party libraries. There are two plugins specifically for SVG, DrawSVGPlugin and MorphSVGPlugin.

GSAP is an actively developed library with an active community. GSAP does not have a native UI pack of pre-set transitions like Velocity, but there is one by a third party called QUI, made by Quasso.

There is also an editor for GSAP as a Google Chrome extension. It is still an alpha release. I did not manage to get it working, but the idea of it seemed very handy. First the animated page is opened in Chrome and the editor opened. Then the animating should be done selecting the element and changing its properties and setting states on a timeline. I do not know whether the editor is able to animate SVG elements, because it is not mentioned on their site and I was not able to test it.

2.21 Activity Tools

There are too many tools that can be used for creating or developing activities to list them here. I will not go into much detail describing them, but I do describe what types of tools can be beneficial.

When creating custom activities from scratch, I have found jQuery to be a great help. It has a great number of plugins from general to very specific purposes. It is recommended to always search for a plugin for whatever new feature needs to be developed. Of course there are numerous similar DOM manipulation libraries available too to choose from. A developer can also benefit from using a HTML5 UI framework, which should be used anyway. There is also WYSIWYG and other editors available that can help in the activity development. These are all tools that apply to typical web development too. Tools that are targeted especially for creating e-learning content are often referred to as authoring tools. The most popular are Adobe Captivate, Articulate product family, and Camtasia Studio [68].

This chapter gave background information to understand the setting where and why the research was conducted. The next chapter, Method and Material, describes what methods were used in the study and how the data was gathered.

3 Method and Material

This thesis is part of a development project in MPS Prewrite Oy. Until early 2014 Prewrite used Adobe Flash as the main client-side technology for its e-learning courses. Prewrite's self-developed learning management system (LMS) is called Gimlet. The Flash-based development kit (devkit), Gimlet Flash Player (GFP), has been used as a base for nearly thousand courses to minimize the technical work for each e-learning course.

The parent project of this thesis is the development of an HTML5 based devkit to answer the growing need for mobile support. At the current state of development, Gimlet HTML5 Player (GHP) is limited in features compared to GFP. The development of GFP was run down, and all efforts have been put to the HTML5 devkit for two years now.

HTML5 standardization is not finished and especially number of the specialised development IDEs can still be considered fairly simple. This thesis researches some of the current tools for specific use cases and evaluates their features and suitability for HTML5 based e-learning, from Prewrite's perspective.

The experiences with GFP have proven that standardizing the frequently used activities saves a lot of time when implementing courses. Therefore it made sense to make the similar standardization for HTML5 devkit (GHP). The activities make the first of the two task categories in the study.

Another very common implementation work is the custom animations. A custom animation can be for example a course's intro animation, a chart animation, a character animation, or a content transition animation. Implementing animations is very time consuming work especially if using ineffective tools. With good tools a lot of time can be saved. The animation tasks make the second of the two task categories. The tasks are evaluating and selecting the best tools for three animation types – general animation, chart animation, and character animation.

Both activities and animations are fairly easy to implement in Flash for courses with fixed pixel size and used in desktop and laptop computers. It is more difficult and time consuming to implement the content and functionality for all kinds of devices and displays with HTML5 – especially the animations. To achieve the proper scalability for the smaller displays the UI

needs to be responsive. Responsiveness will be one of the sub topics of the study. I try to research how to implement it in the study subjects.

This thesis studies JavaScript libraries, HTML5 and CSS3 techniques and design patterns for activities and animations, and HTML5 Animation IDEs and libraries. Later these are referred to as just tools.

The study does not cover:

- Tools suitable for other page elements, course elements or exercises.
- Usability issues or user experience from the end users perspective with an exception of issues related to responsive design
- Issues related to customization of the activities or animations
- All tools and technologies related to HTML5 activities and animations – only the specialized tools suited for each specific task in the thesis.
- Linux or Mac specific tools

Prewrite is fairly new to developing e-learning with HTML5, so knowledge on which tools to use for each task category, and how, is needed. The tools need to be easy to use, because the junior developers are not experienced in advanced programming. The amount of time saved with proper tools and best practices in both task categories can be hundreds of hours per year.

3.1 Methods

The research can be divided to two main phases. In short, in the first phase the available tools for each use case were gathered, evaluated, and compared against the requirements and then against each other. Then the tools used were selected based on comparisons. The second phase was to get the proof of concept. To do it, the selected tools were tested by implementing the two activities to GHP and creating test animations, keeping notes of the problems, experiences, and other findings. After the tests, the notes were analysed and final conclusions were reported. Table 1 shows the strategies, approaches, and methods used in the research phases.

Table 1. Used strategies, approaches and methods in the research phases

	Phase 1.	Phase 2.
Strategy	Mixed	Qualitative
Approach	Evaluative study, comparison	Tests
Method	Analysis of data	Solution development

This thesis used a mixed research strategy, i.e. both, qualitative and quantitative research methods. The idea in phase 1 was to find the most efficient tools for the both task categories by comparing the tool features and qualities to their requirements. They could not be measured with all absolute and standardized values. This made the used variables and values non-scientific because they were not all exact values, but subjective evaluations. The used research approach was evaluative study.

For proof of concept the tool sets were tested by developing two standardized activities and an animation of each animation type. The qualitative method was used to analyse the results of the tests.

3.2 Process

This thesis is a case study that uses a linear research design. First information was collected, analysed, and structured. As candidates were found they were also dropped in a preliminary screening, if they did not meet all the mandatory requirements. Then an evaluation and comparison of the tool candidates was made and the final set of tools was picked. Finally the tools with top scores were tested and the test results analysed.

To know what kinds of tools were needed for tasks in both categories, a list of the technical requirements for the tools was made. The selected tools should fulfil these requirements. Also two activity types were picked to be created for testing the selected tools.

To find and to be able to evaluate the available tools or tool sets to be used, source material was researched. Studying library sources and Web enabled making a list of the tools available used in HTML5 development suitable for both task categories. It also provided information about their features and qualities and what they are used for.

To compare the tools, the tool features were evaluated and comparison tables with the essential requirements and values from the tool evaluations were created. More about the analysis is in Chapter 6, Data and Analysis. At this time the final tool set to finalize the tasks in both categories existed.

Finally the tools were tested by creating the two activities selected for testing and a test animation of each selected animation type. Notes were kept on the problems and other relevant observations during the development. Then notes and the result were analysed and reported how the selected tools apply for tasks.

3.3 Data

The information gathered for the evaluations is mostly not scientifically produced, because it is frequently from non-academic Web sources. Reasons to this could be that the subjects of the study were too marginal or too new to be included in any academically credible materials. The data was mainly scattered on numerous web sites. Assumptions can be made that this information can be occasionally even bias opinions, but this is difficult to validate. But it was still the best information available. The data is collected, analysed, and structured information about

- the tool candidates that could be used for the tasks
- the feature requirements of specific activities and animations and in general level
- and the experiences from the tests creating activities and sample animations

Feature requirements for the activity and animation tools were collected: from company documentation; from the developers who uses the Flash versions the activities and create the animations; from researched literature and web sources; and by studying the functionality of the activities and animations. Only some data for the requirements was therefore gathered from academically reliable sources. Some of the data for defining the requirements is from my personal experiences. In the study, I have tried to justify using them case by case.

I read through a great amount of literature on the topics for the study. Academic and other library sources provided general information for the background chapter, but only for a few tools for the evaluation. The tool data for evaluations was collected primarily from the developer product sites or GitHub pages. If information was not available there, a third party web shop or a review was accepted as a source. If information could not be found at all,

tests were made with the tool to produce the data for the evaluation. The reliability of the tests can be questioned, but less in the tests where the tool succeeded the given task. In the failed test there is an increased possibility of doing the test wrong.

An assessing system was created to evaluate the tool properties and score them for the comparisons on an equal scale. If evaluated property had a numeric value, I set a minimum or maximum value for each score from 0 to 5. If property did not have a numeric value, I tried to convert it to a number or create another system to rate it. The tools and their features differ from each other so that it was occasionally hard to rate them on the same scale. Therefore there is a possibility of me being unconsciously bias when rating the property data.

The developers, mostly me, kept notes while developing the test tasks. Notes include the experiences of the developers like problems that came up and other relevant observations on using the new tools.

I used data and information partly published by others and partly created and verified by myself. Data from others, but collected by me, is the list of the tool candidates and information about their features and properties. Also the notes from the final tests of creating the sample activities and animations are data created by me and others. Data purely created by myself is the evaluation and comparison tables and their results. No conflicting data was found in the study. I ensure my own descriptions, interpretations and conclusions correct and honest.

The next chapter, Tasks, defines the research tasks the study should answer.

4 Tasks

Now that basics of the GHP have been cleared, this chapter defines the task subjects. It also sets the requirements and describes the tools that qualified to evaluations and reasons why some of the tools did not qualify. The tasks are divided to two categories, activities and animations. The reason for choosing these categories was because implementing custom solutions of either category usually consumes a lot of work. Standardization, templates and the right tools will significantly decrease the time consumed using features in the courses. The purpose of the tasks is to find tools or technologies to perform specific tasks with specific requirements. The evaluated tools or technologies will later be referred to as just tools. The subsection for each task describes the features the selected tool should help to perform, and what are the requirements for the tool. They also list the tools that qualified and the reasons why others did not qualify to the evaluation.

There are requirements that are common to all tasks. They are described here. To have the proper scalability for the smaller displays the UI needs to be responsive. Therefore responsiveness is a requirement for the tasks. Also all tools with an editor need to work in Windows. Therefore no Linux or Mac specific tools are evaluated. Naturally the tools need to support all modern browsers including IE 9.

Tools should be actively developed to get support, necessary bug fixes and to ensure the support for future browsers. Also free licence is preferred. Tools can have a paid licence, but it needs to be reasonably priced for our usage. If separate fee is required for each course or instance where library is used, it will probably be too expensive to suit the purpose.

For all libraries a small file size is preferred. Smaller file size does not only decrease network traffic, memory consumption, and execution time [9,484], but only expands battery life. The tools get also better scores in the evaluation for active development, good documentation and examples in the web site, and active community to get support from.

4.1 Activities

The definition of activities was covered in Chapter 2.7, Activities. The tools sought in the category are all JavaScript libraries. They may come with supporting CSS files and even some small media assets such as icons. This category has only two tasks, because there

were only two activities missing from GHP that could gain from a third party library. GHP has a Tabs and a video player activity that could have benefitted from an existing library, but they were already implemented. In video player we use Video.js and have been satisfied with it. Tabs use only custom code.

All tasks in this category share a number of common requirements. They are:

- Easy visual customization with CSS
- Text formatting for content
- Support for custom element sizes

The task specific requirements are described in the following subsections.

4.2 Hotspots

This is a common activity that can also be found used outside the e-learning world across the web. So it should be reasonable to assume there is a third party solution for implementing this. It is used when additional information needs to be linked to specific parts of an image. The elements of the interaction are:

- image
- the spots that are the active areas on the image
- tooltips that appear when a spot is activated

Examples of the Hotspots elements can be found in Figure 6. It presents a screen capture of a full page Hotspot activity from a demo course.

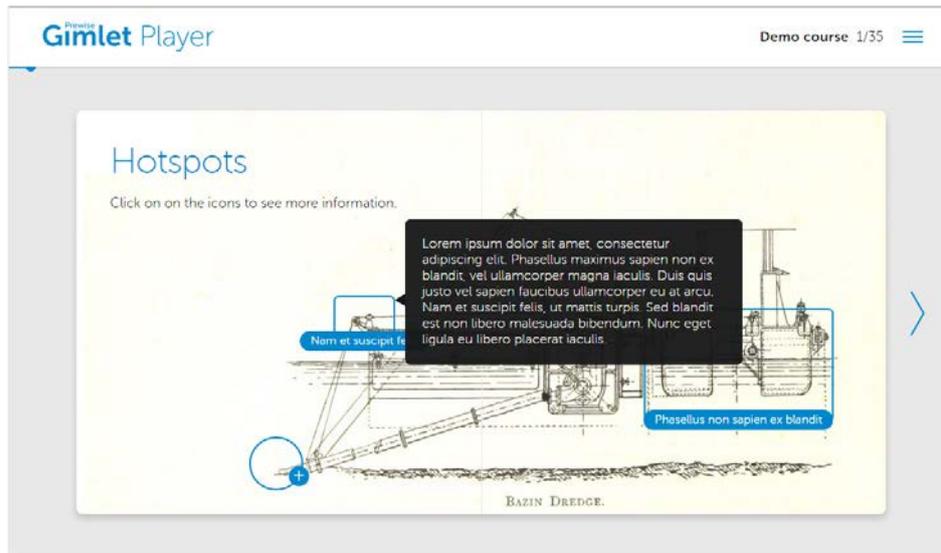


Figure 6. Full page Hotspot activity.

The image element is in the background in Figure 6. The spots are the blue elements that mark the hotspots by outlining the active area. The tooltip is the dark rounded box in front of other elements.

The first of the activity category tasks is finding a solution that does the Hotspots functionality as effective as possible and meets the requirements set to it. The solution is probably a JavaScript library or a combination of several libraries.

The hotspot areas can be placed anywhere on the image area and the number of items should not be limited. As default the area is a ball, but it can also be set as a rectangle. The width of the area can be set and for rectangles also the height is required. The trigger event for showing the bubble can be hover or click. On mobile it is always tap. The areas should have active and inactive state. It should be possible to set transparency values for the area's inactive and active states. An optional short caption text can also be added to an area. The spot needs to be easy to style to client brand requirements.

When the user activates an area, a tooltip opens and shows the additional information to that specific area. The tooltip is a box containing formatted text. It can also include images. Tooltip should also have a tip or a stem pointing to the area it is related to.

The tooltip should grow to fit its contents. The tooltip should change shape and position so that it always fits to the page. It should be placed very close to the icon and placed to the

direction that has most free space to contain it. This means that the tooltip should have at least 4 positions for the tip: top-left, top-right, bottom-left and bottom-right. Even better would be additional right-top, right-middle, right-bottom, left-top, left-middle and left-bottom. The best would be if the tip could be placed to any position on any four sides. The tooltip needs to be easy to stylize.

In mobile, areas are tapped and tooltip hides when something else is tapped or the icon is re-tapped. Tooltips should have 'show' and 'hide' animations as response to action and to enhance the user experience.

These can be challenges developing Hotspots. The spot area size, on the image, needs to be defined as percentages of the image width and height. This is because in HTML5 player the image size changes with window size as being responsive.

After a day of searching for libraries and tools for Hotspots, only two libraries were found that could possibly meet the requirements for Hotspots. They were Nicky's Hotspot Map and LiteTooltip.js.

Nicky's Hotspot Map

This was the only full library specifically targeted for making hotspots that had some advanced features that were required. The speciality of the library is that it comes with an editor to create and adjust the settings for the hotspots. The library was not perfect for the job so more contestants were needed.

LiteTooltip

This was not specifically a hotspots but a tooltip library. Still it included a feature to create hotspots. It gave a very professional feeling overall compared to Nicky's Hotspots Map. LiteTooltip is a jQuery plugin with good mobile support and documentation.

4.3 Carousel

The purpose of the second task in in the activity category is to find the best tool available to add a 3D carousel feature to GHP. This is an activity that did not exist in GFP, but was decided to add as the tasks for the thesis were planned.

A carousel is a loose definition, but common to them is that they always consist of multiple pieces of content [69] – let us call them slides. And the slides are seen in a space, one or multiple at a time. One of the slides is selected or is active at a time. That one should be clearly visible so that its content can be studied easy. Often the slides preceding and following the current are positioned to its sides and partly covered, faded, or tilted so that they are less visible. The slides are linked to each other as a strip so that as one is moved, the ones next to it move along as seen in Figure 7. The strip can be never ending, so that after the last slide comes the first one. Carousels found in the web are more often just “2D” versions where the slides move in line on a plane. The tool sought in the task should also support 2D modes.

Sometimes accordion type activities are considered as a form of carousels [69], but I feel they are too different to be counted as carousels. First, parts of all their slides, the title bars, are visible at all times. And second, they do not have the same appearance as chain of slides rotating, because their parts hide very different from a typical carousel where slides slide off the carousel viewport or move to the sides and to the background in the 3D carousel.

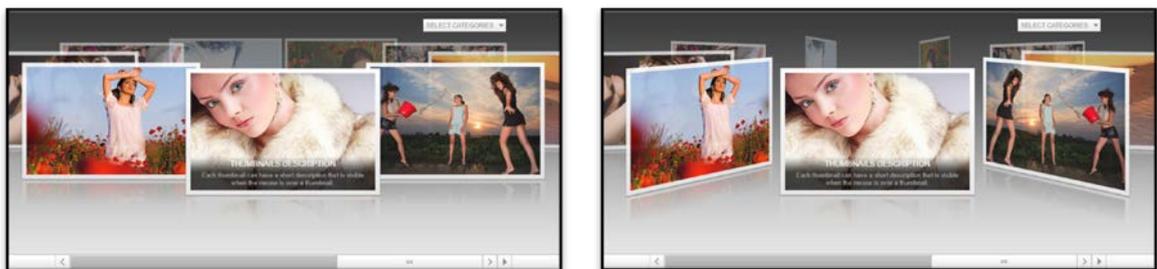


Figure 7. Examples of a typical 3D carousel on the left and a star shaped on the right from a demo site of Ultimate 3D carousel by Future Web Design [70].

The main reason to use a carousel in general is to enable multiple pieces of content occupy the same space on a web page where all space is valuable [69]. For the use in GHP there is an unexplored possibility to take advantage of carousels linear form. Because all the possibilities how to use this activity type are not yet established, it is better to prepare for formatted text, images, animation, interactions and videos to be used as the content on the slides.

In this task the aim is to have an option to render the activity as a 3D carousel. The illusion of a third dimension can be achieved with several visual effects such as shadows, lightness changes, blurring, perspective distortion, and positioning element behind and in front of each other.

Carousels are often used as so called marketing carousels to lift and promote specific items on a front page of a site, such as news items, products, or popular TV-shows. They are also used as image galleries for example to show product images in a web shop. There are many other use cases as well.

4.3.1 Requirements

There has been quite a lot of discussion, firmly supported by studies, about whether the carousels should be used at all, because of their usability issues [71]. A study by Jacob Nielsen shows that one mistake using a carousel is having it change slides automatically [72]. Derived from the study results user control should be a requirement. Controls should also be made obvious so they are not missed – a common mistake and a requirement for the tool as well [71; 73; 69]. It is suggested too that carousels would clearly indicate the slide count and current position in it to let user feel in control [69].

After going through Nielsen's and Erik Runyon's studies showing how heavy the click focus is on the first slide [74], I concluded that these studies do not fully reflect the use case in GHP. I see it having more in common with the use cases in a study by Kyle Peatt and Peter Maclachlan. They had positive results in the use of carousels with product detail images in web shops. In the studies against carousel usage, the use cases were web sites having lots of information visible at a time and sites with subpages, information and links. A page layout in GHP is very simple compared these cases. Actually the carousel will be the only content element on a page in some cases.

Also in the studies against carousel use, the carousels were used to promote independent topics or pages as large and advanced ad banners. In GHP use case, the slides will have content from the same topic and in some use cases even form of a story, taking advantage of carousels linear form. Also the user knows all content needs to be read and completed so we can assume there will not be a problem user not seeing all slides. Going through all slides can also be made a requirement to move forward in the course.

Activities such as carousel are not always intuitive to all the users. To hint them about gesture controlling, there could be a little initial animation to provide a clue [73; 69].

In general 3D Carousels take quite a bit of horizontal space compared to vertical space consumed as seen in Figure 7. This can be affected with the geometry of the carousel. This meaning the angle, shape, size, position, and distance of the slides from the centre of the carousel. In Figure 8 find a ring shaped and a twisted ring shaped carousels that take less horizontal space visually than the typical and the ring shape in Figure 7 before.



Figure 8. An example of a ring shape carousel [70] and a twisted ring shape carousel [75] from Future Web Design.

It will be a requirement that the carousel should support different shapes to fit the carousel in a responsive page and also to have an option for variation.

To use the carousel covering only half of the player's content area, it should support vertical direction too. In vertical mode there will be space left for body text, but the 3D effect needs to be removed for placing the slides to fill the smaller area better.

4.3.2 Risks

Carousels are complex in many ways. For a user as discussed, but technically too. They require more code, include animation, and in our case options for different render types, which all increase the possibility for malfunction [73]. The used library should be well crafted, tested and supported for good browser and device compatibility. Testing quality of the libraries is very hard to evaluate without thorough testing program. There is no time for it, so the testing score is based on the overall image from the materials related to the library and the experiences in test made with it. And also what others have written in reviews and developer forums. Testing quality would deserve more weight on the final score, but due to difficulty of evaluating it, the weight will be less significant.

We might also face challenges with responsive behaviour. Carousel should change from 3D mode to 2D with smaller displays.

4.3.3 Tools

As mentioned earlier, the definition of the term carousel is loose, but even with a strict definition there are tens of libraries available to create them. They vary from simple to advanced ones and from very poor to fairly good.

Finding the libraries to evaluate took lots of googling and going through pages listing '45+ Best jQuery Carousel Plugins' and 'jQuery Carousel Plugin & Tutorials with Example'. I only needed three search phrases, though, to find the candidates for the evaluation. These were:

- "3d carousel JavaScript library"
- "css3 3d carousel"
- "gsap 3d carousel"

For each search I went through the links from the first two pages, skipping the obvious off topic titles. A number of these links were listings of more links which I also investigated. I estimate finding and going through 80 libraries. Some I managed to discard faster than others. Rejections were made on how well the library's properties met the task requirements. The desired 3D look eliminated probably 80 percent of the carousel libraries. Find more information about how the libraries were gathered and how they were screened in Chapter 3, Method and Material.

Next follows information of the top libraries left outside the evaluation and why. An HTML canvas based library called HTML5 Canvas Carousel was dropped out from the evaluation, because the canvases lack of support for HTML formatted text. Circular Carousel is very small in file size, but poorly documented. It was dropped, because it was missing support for IE9 [76].

Storyline 3D Slider seemed very suitable in quality and features. It was missing the vertical mode, but the worst was that the licensing was per client. The licence would have had to

been paid each time we used it for a new client [77]. This would have been too complicated. Nobody would have remembered to pay the licence when the time came.

JQCarousel seemed very good in the beginning. It even had an option for vertical rotation. With JQCarousel I tried to test its scaling features in JSFiddle, because there was no mention of such on its web sites. I could not get the library working. There was a JavaScript error. I decided to drop JQCarousel from evaluation for being poorly documented and tested and giving too little promise of proper features and quality. It was also lacking touch support [78].

JQuery Carousel Evolution has a great set of features, but has not kept up in the requirement of the modern web. The last update was from 2011 and the library was missing support for auto scaling so it had to be left out the evaluation. Scalability was tested in a test with JSFiddle by me.

Flipster, Killer Carousel, and Cloud 9 Carousel both have extensive settings and features and good documentation. The only major defect of the libraries, and the reason for dropping them, was lack of support vertical mode. This was confirmed in my tests in JSFiddle.

Future Web Design (FWD) is a Romanian company or group selling various jQuery and WordPress plugins. The quality of them seems very high by examining the marketing pages, demos and feature lists, even better than the libraries selected for the evaluation. FWD has three jQuery plugins that all appear highly potential for the evaluation - Simple, Royal, and Ultimate 3D Carousels. The libraries seem to be based on the same engine but they all have different features from each other. These libraries were dropped from the evaluation because of their unclear licensing fees. I made an inquiry about them to the developer, but did not receive a reply.

CarouFredSel

Despite the strange name this is an efficient library, though it has been last updated three years ago in April 2013. What are especially good about this jQuery plugin, are the 72 examples found in a gallery made by the original developer. The plugin was acquired by Dev7Studios.com who sells it as a WordPress plugin. It can still be used just as a free jQuery plugin with help from the extensive documentation by the current developer. The library does not have as well built-in features as its top competitors, but it is very customizable and extensible as the 72 different examples proof.

Swiper

Swiper is a library by iDangero.us. It is designed to be used in mobile websites, web apps and native/hybrid apps. It is useful that they have made a lighter version of the library to be used with jQuery or Zepto. To make this library better for GHP use case I would enhance the 3D effect and add more shape options for the 3D carousel. Enhanced 3D effect could be achieved by fading the slides, the further they are. Swiper's only 3D carousel shape type is none of the common shape types, so I would add them to the library if possible.

4.4 Animations

Another very common implementation work type is custom animations. They can be the course's intro animations, process animations, character animations, content transition animations and so forth. Implementing animations is very time consuming, especially if using ineffective tools. A lot of time can be saved with proper tools. This thesis tries to evaluate and select the best tools to create three animation types for GHP.

As mentioned earlier, the animation types were defined by grouping animations with distinctive features and similar requirements for the tools for them. The animation types for the research were selected based on assessments how likely there could be specialized tools for them.

The selected animation types are general animation, chart animation, and character animation. Animation types that were considered but dropped from the research were wow-animations and screen captures. Wow-animations are not a very consistent type and the tools evaluated would have been very different from each other in terms of features and therefore difficult to evaluate. The screen captures on the other hand have almost a de facto tool, at least in e-learning, Adobe Captivate. It felt unnecessary to start evaluating other tools against it.

The animations will not be responsive as default, but they should scale. They might be considered responsive in terms of text in them. It should be possible, and recommended, to increase the text size in animations on smaller displays with CSS.

The tools and technologies already selected for GHP and its development also set requirements for the ones evaluated and selected in this thesis. The tools and technologies being

selected cannot conflict with the ones already used. It is even better if they utilize tools already used or vice versa. The tools should support vector graphics – that meaning SVG in HTML5. This is because SVG is very light weight, the elements in it are animatable, and animated graphics in the categories are often created originally in vector form, in our case with Adobe Illustrator.

The following subsections explain the requirements for the tools of each animation type. They also include explanations why they were selected as a requirement. The subsections also provide a list of tools that qualified for the final evaluation and also explain why some tools were dropped from the evaluation.

4.5 General Animation

The animations that fall into this category can look very different from each other and have great amount of variation in the philosophy how they are created and structured. There are tools for animating elements inside a SVG, but many of them are for some specific type or types of animation. The tools that are developed for a specific type of animation will not be included in the evaluation. This task focuses to a generic type of animation.

The selected tool for this type should be very easy to use for this purpose. The most important requirement for general animations is transitions. Transitions animate the shape, size, position, angle, colour, and opacity of an element. These animations can be long so it would help if there was a concept of a timeline for timing the element's animations. Also the tool should be able to animate between two values of elements properties, resulting to a requirement of keyframe animation support. These requirements are best met with a WYSIWYG type of editor if one would be available. This also means that all frame animation tools are excluded automatically.

All evaluated tools need to be compatible with the vector graphic tool our designers use, Adobe Illustrator. This making a requirement that graphics made in Illustrator have to be importable to the animation tool.

Wishes are also that tool should also support setting call backs in different states of the animation, the most important being on end and on update. Also animation should be controllable with commands such as pause, resume, and seek. Good performance, support for

transform origin manipulation and support for morphing shape and line result to better score in evaluation.

Tools

Adobe recently released new Web Animation software Adobe Animate CC that will replace Adobe Flash Professional as their flag ship animation software. It renders as canvas for HTML5 support as default. It supports SVG animation through an add-on called Snap.svg Animator. The greatest disadvantage of using it is the large generated file size. It saves the element states for each frame meaning it does not support keyframe animation. In my test for animating one elements shape and others position and rotation for 2 seconds between 2 elements states resulted to 200 KB of file size. With keyframe animation it would be probably less that one 10th of that. The advantage of saving the state for each frame is better performance. It does not require resources for calculating element states between the key frames. I still feel that it generates way too large a file in relation to the number of elements and length of the animations. Also the add-on does not yet support any scripting, font or image embedding [79], but the main reason to exclude it from the evaluation is lack of keyframe support.

Adobe has their old HTML5 animation software Edge Animate CC. I decided to exclude Edge from the final evaluation as well, because Adobe has announced that Edge is not developed active anymore [80].

Inkscape is a fairly popular free open source SVG editor, but it does not produce proper SVG animations even with its plugins, so it is not included in the evaluation.

Animatron seems like a great online editor but it outputs SVG SMIL animations which are not supported in IE so it does not fit the requirements.

CSS would be a handy and light-weight technique to animate SVG. But it does not work on IE properly [81]. IE does not support CSS transform that are essential for scaling, moving, rotating and skewing. So CSS is not included in the evaluation.

Snap.svg was already covered in Subsection 2.19, SVG Tools. It has SVG animation methods, so it could be included in the evaluation. The animation methods are somewhat limited though. And I know this from experience, so I decided to leave it also outside the final eval-

uation. If one ends up using Snap.svg and GSAP in the same solution, a plugin might be wanted to use GSAP animation methods in Snap.svg. The plugin is called GSAPSnap-Plugin and it is made by Anthony Greco. Svg.js is also an SVG manipulation library, but excluded due to very limited animation capabilities.

The libraries that made it to the evaluation are Velocity.js, Popmotion, and GSAP. GSAP was already covered in Subsection 2.20, General HTML5 Supported Animation Technologies.

Velocity.js

Velocity.js, later just Velocity, is a light-weight HTML5 animation library made by Julian Shapiro. It has a good support for SVG animations. It was made as a jQuery plugin to replace jQuery's own animate method, but works nowadays also without it. JQuery's animate method is not compatible performance-wise. Velocity has an additional library UI Pack to add pre-defined transitions. I feel this is a useful addition. Timeline support can be added to Velocity with third party library called Tweenie. Tweenie can also be used to add additional features to jQuery, Transit and GSAP.

Popmotion

This is a new comer as an animation library. It was released less than a year ago in May 2015. Its specialty is a method for physics simulation.

4.6 Charts Animation

The second animation type is charts. This is a type of graphic we often like to animate appearing on a page to make it more appealing. The data oriented infographics tend to bore people and we try to avoid that making them more engaging. This could be done using the same tools as the general animations, but I know there are JS libraries that render and animate graphs from data structures given to them. This seems like a smart way to create them from the data and set the required properties for how the chart should look. Those should be the three main requirements: renders from data structures; easy, but extensive visual customization and has built in animation features.

Because there are so many options in chart tools I also came up with additional restrictions to narrow them down. The tool should only require other frameworks or libraries specifically targeted for charts. Any general purpose libraries would be only be partly used, but would still increase the byte size loaded to browser. Also the licence for using the library should be preferably free, but paid libraries can be included in the evaluation if the price is fair. The tool needs to be actively developed. Inactive projects do not get bug fixes and support. This requirement is stricter than it was for accordion tools to filter out more evaluation candidates. Tools that have not had a release in a year are excluded from evaluation.

Although the task is to find a powerful chart animation tool, the tools also get points from additional interactive features empowering the user to explore data for themselves. The selected tool should support as many chart types as possible. But at least the following:

- Column or bar
- Line
- Pie or donut

We need to use the client brand colour palette in the courses so also colours of the charts need to be customizable. The chart axes need to show the name and unit. There should also be support for data labels and legend that show which colour refers to which data source. 3D effects in the charts are an optional requirement giving extra points in the evaluation.

This task would have a lot more options to choose from without the SVG restriction. Canvas provides a very good option to SVG as a graphic format for drawing and animating charts.

Tools

Before going further with charting libraries it is necessary to point out that one library dominates the data visualization library field, D3.js, later just D3 as in Data Driven Documents. It is a low level open source library for generating and manipulating SVG visuals. It does not have too many predefined visualizations or chart properties compared to others. Its strength is in versatility to bind data to its visualizations and removing browser inconsistencies [62,45]. It has a steep learning curve. To overcome this, there are 19 data visualization libraries listed on their web site taking advantage of D3 and in a way making its use easier. Some of them are listed below.

jQuery has a lot of chart plugins. Only some of them are included, because most of them are mainly for certain types of charts and we are looking for a comprehensive solution.

JenScript had potential, but most of the demos on its web site did not work, so I decided it is not worth including it to the evaluation [82].

Plotly.js is built on D3 and Stack.gl for WebGL 3D. It comes with an online chart editor and seemed like a great library, but the demos on their page did not work on IE9, so I had to exclude it from the evaluation.

Online Charts Builder is an online tool for creating charts with an editor. The tool generates an iframe element to be embedded on a web page. The page loaded into the iframe uses Google Charts to render the graphics, but it is possible to extract the script and place it directly to a web page without the iframe. The tool will not be part of the evaluation, because it has not been active for a year [83]. It could be used though, if Google Chart Tools would be selected as result of the evaluation, because editor will make it easier to generate the JavaScript creating the custom chart.

ChartBlocks is an extensive online chart editor. There is a moderate monthly fee for advanced usage. It is excluded from the evaluation because charts are not animated at all. Peek is a D3 based chart library excluded for the same reason. Online Charts is a similar service, but exports plain bitmap images, so cannot be animated and not included either.

Rickshaw and xCharts are a similar D3 based libraries. They are excluded because they do not support pie or donut charts [84] [85]. Raw is a D3 based library that comes with an editor, but does not support any of the required basic chart types, only special or advanced ones [86].

Protovis is an extensive chart library. Unfortunately the developed has stopped. The developers have joined the D3 team. The library is not in the evaluation because it is not developed anymore. [87]

Morris.js is a lightweight chart library based on jQuery and Raphaël [88]. The library was excluded from the evaluation, because it requires an external library we do not currently use, Raphaël. Same applies to Elycharts, gRaphael, ICO, and Grafico. Vaadin Charts requires a polyfill for Web Components so it is excluded as well.

Sencha Ext JS web application framework has a comprehensive chart library. It was left out from the evaluation because of its high price starting from \$4,340 and because it comes with a web application framework we do not need. [89]

Ember charts also rely on a framework, Ember.js. Dojo Charting instead requires the Dojo toolkit. YUI Charts is a widget for YUI by Yahoo. JointJS chart widget needs the JointJS framework. All excluded from the evaluation for requiring an additional unnecessary library.

The following promising libraries were dropped out from the evaluation, because they rely on canvas for drawing the chart: Dygraphs, Flot, Chart.js, jqPlot, jQuery-Visualize, JS Charts, rgraph, canvasXpress, JavaScript InfoVis Toolkit, AwesomeChartJS, SigmaJS, ZoomCharts, Enterprise JavaScript Charts, Online Chart Generator by LiveGAP, arcadia-Charts,

ZingChart and amCharts were dropped out from the evaluation because of their high price for the licence we would require. They are both powerful libraries. AmCharts might be the best chart library I ran into in general. It had the most advanced animation features and an editor so charts can be created easily without any programming.

JQWidgets is rather expensive, but it has a decent online chart editor. The main issue why it is excluded from the evaluation is the lack of support for custom colours.

Chartist is a small library that was developed for creating simple and responsive charts [90]. For a library of only 10 KB it seems very good. The only reason it is not included in the evaluation is because it uses SMIL for animations. And SMIL is not supported in IE. NVD3 is a good basic library, but it does not support IE9 either [91]. Same goes to Factmint Charts and otherwise very promising UvCharts based on my tests.

Highcharts comes with an editor. It seems very handy for setting up the properties for wide range of charts based on their marketing video. Highcharts cannot be used because data privacy issues. It is a cloud service where the data is saved to cloud and chart is included as an iframe to a web page.

The following tools and libraries got pass my preliminary screening and are the top 13 chart tools. There was still too many for all of them to be evaluated, because evaluation is very

time consuming. So I needed to further narrow them down. I was having hard time deciding the criteria, but ended up using their automatic animation features and capability for custom animations with settings. I looked through the API documentation and examples on their web site. If I was not able find the information there, I reviewed googled examples and Stack Overflow answers. Both criteria were evaluated on scale 0 to 5. Only one library got over 4 points total. It was C3.js with 5 points. Four libraries shared the second best place, all with 4 points. These libraries were Plottable.js, Dc.js, jChartFX, and Google Chart Tools. 4 points out of 10 is not much. Because the libraries seemed all very capable otherwise, I made a judgement call to put great weight on the animation capabilities. I decided to set the minimum requirement for qualifying to final evaluation to 5 points. This was a search for the best animation library after all and all libraries that got this far seemed otherwise efficient enough for GHP. This meant that there would be the only one tool to pass for final evaluation. The following libraries were excluded from the final evaluation based on their features to animate charts.

As significant a data visualization library D3 is, it lacks automatic animation features. With custom scripts it is possible to do pretty much any kind of animation though. Plottable.js, Dc.js, Dimple, and D3plus are free D3 based chart libraries. Dc.js is a multi-dimensional charting library built to work natively with crossfilter.js, enabling working with large datasets. Dimple is aimed to make it easier for analysts to use the power D3. D3plus is an extensive chart library with a built-in text wrapper.

FusionCharts is a very extensive paid enterprise-grade chart library. It has 90 chart templates ready to use. It has good documentation with articles and examples [92]. Charts seem easily customizable, but have poor animation features [93]. JSXGraph, AnyChart, DxChart, Kendo UI Charts, and Shield UI Charts are all similar good paid libraries. JSXGraph is a Mathematics visualization library. AnyChart is a powerful and extensive chart library. DxChart by DevExtreme and Shield UI Charts are jQuery plugins and parts of a larger paid widget library. DxChart has extensive features and good documentation. Kendo UI Charts is part of Telerik's UI widget gallery, but can be used as such without other Kendo libraries. JChartFX had better animation capabilities than the paid libraries already mentioned, but still not good enough.

Google Chart Tools API is a comprehensive data visualization tool set. It can be divided into two parts. Image charts are rendered as an image in Google servers from the data provided for them. The other part is Interactive charts. It is a library loaded to browser. We

cannot use Google Image charts for issues in data privacy. We cannot have our client's data be sent to Google servers. But I could have included the library to evaluation on the parts that do not need sending data anywhere.

4.7 Character Animation

Character animation was selected as a category, because, in my experience, it has special requirements from the tools.

It was earlier covered that character animations can be used as a way to apply the Personalization Principle and use them as Pedagogical Agents. I feel they could also be used even more, for a similar purpose. Tversky et al. is concerned that animations only show, but do not explain enough [20]. I suggest characters could be used as part of an instructional animation to be the entity doing the explaining. This would be done with a voice narration. This said – support for audio synchronization should be one requirement for the character animation tool.

It would be very convenient to find a lip synchronization (lip sync) tool for HTML5 animations. There is at least one for Flash, SmartMouth, which is given images representing 8 mouth expressions that cover roughly all the main speech sound expression (phonetics). Then it analyses and interprets a speech audio file and based on the sounds spoken on the audio, it arranges and times the mouth expression images on a timeline to visually imitate a speaking mouth [94]. Based on my experience the animation is quite convincing when the timeline is played with the audio. If there are no lip sync tools for HTML5 development, I guess it could be possible to use the flash plug-in and export to HTML5.

A feature I missed at Flash 8 was a way to connect body parts together with rotating “joints”. This would make animating a character a lot faster. Conceptually the difference to average animation is in the hierarchy of the moving elements. Usually a movement of an element in an animation is not dependent on a movement of another element. For example cars move on a road independently. But in a character movement of an eye in the whole animation is dependent of the movement on the head. The movement of the head is connected to the position of the torso, and the movement of the torso to the feet. I suppose one way to implement connected joints is by creating a hierarchy of grouped elements in SVG. If there are no HTML5 tools available for inverse and forward kinematics, it could be done in Flash 9 or greater and trying export it to HTML5.

Although the task is to find the best library for SVG character animations, I feel I should list other libraries that suit other HTML5 technologies. Canvas has different game engines built for it. One of them I ran into is EaselJS. It is part of a suite of modular libraries by gskinner. In EaselJS the character animations are created with looping frames in a sprite sheet [95]. Looking at the demos this seems very easy way doing it and works well too. I bet by re-searching them, one would find several Canvas tools suited for character animations. But because canvas is pixel graphics, it does not suit well enough to our needs.

Crazy Talk Animator seems really extensive in features. Its features include human inverse kinematics, lip syncing, 3D characters, and facial puppeteering [96]. It seems suitable for TV animation productions based on the documentation and demo videos. Crazy Talk Animator does not meet the main requirement for format, because it does not export SVG.

When it comes to SVG tools, I could not find any tools especially for character animation or with features especially for it. Find more info about the results in the next chapter, Results and Analysis.

5 Results and Analysis

This chapter explains the results of the study and tries to analyse them. First this chapter follows a similar structure as Tasks with own subsection for each task. These subsections reveal the results with analysis of the results for the task. The end of the chapter has analysis of the results on a more general level.

After collecting and analysing the information on the tools, I was able to grade the tool features. I gave numeric values, on scale from 0 to 5, on how well each tool fulfilled each feature requirement. I was then able to compare the total tools points against other tool scores and see which tool was the best. These evaluation tables are presented below.

The report, based on the development notes, points out the most relevant of each selected tool, analyses the experienced problems and questions some of the tool choices.

5.1 Hotspots

Evaluating the Hotspot libraries required acquiring the missing data with tests. The tests were made in JSFiddle developing a done that would provide information whether specific requirement was met or not and in what extent. LiteTooltip.js won the comparison with 208 points to Nicky's Hotspot Map's 183 points. The difference in the score was about 10 % so it was clear, but not overwhelming. Since the evaluation was made they have changed the name from Nicky's Hotspot Map's to Image Map Pro and made changes to it. The scores given for each evaluated feature and property are presented in Table 2.

Table 2. Evaluation data for Hotspots tools [97] [98]

		Nickys Hotspot Map	Lite Tooltip.js			
Requirement	Weight	Tool 1	Tool 2	Legend		
				Value / As colour percent		Verbally
Placeable areas	5	3	5			
Area size	5	3	3	5	100	Perfectly, entirely, very much so
Area states	3	4	2	4	80	Yes, but not entirely, pretty much so
Area events	3	4	2	3	60	to some extent
Area transparency	2	4	4	2	40	some, a little, not much
Area label	2	0	2	1	20	very little
Area shape	4	5	4	0	0	Not at all
Tooltips or co-operation with custom tooltips.	5	4	4			
Areas easy to customize	4	4	4			
Easy to use	5	4	4			
Mobile support	4	2	5			
Tool tip hides on mouseOut / retap	2	5	3			
Places tooltip to optimal position in content area	2	1	2			
Tooltip animations	2	3	3			
Multiple stem positions	3	3	5			
Good documentation	3	2	5			
Free and friendly licence	2	2	3			
		183	208			

The LiteTooltip would have suited best for Hotspots, but the licence was unclear whether we needed to purchase the licence once or for every instance we used it in. The licence is per product and someone could see the courses as their own products. Because of the unclear licence we asked about it from the developer, but we did not receive any answer to our enquiry about the licence. This was a drawback and the evaluation of the hotspot libraries felt afterwards unnecessary, because we decided Nickys Hotspot Map was not advanced enough when the evaluation was done. We still felt we should definitely use a library for the tooltips in Hotspots. We decided to use the same library that was evaluated best for Dialog activity, qTip.

Because the third party library did not render the hotspot areas, we needed to write code for that our selves. This was not too hard but there were some parts that were not straight forward.

The first problem was that the circle shaped area has a relative width to the image used on the background. The height cannot be the same percentage as the width because then circle would always get the same relative proportions as the image. Therefore the circle would not be a perfect circle but an oval. The perfect circle was achieved by placing a div element inside the circle with padding-top of 100%.

The second problem turned out to be centring a label to the centre of an element and making sure the text length would not cause problems. This required a couple of more div elements and some CSS styling, but was achieved.

The "installation" of the plug-in was fairly easy. We added the plug-in to our project sources with Bower. With Bower updates to the plug-in are automatic. The library's JS file was included to our code with requireJS. The hotspot areas, balls and rectangles are generated with a knockout and data is bound to the elements from hotspots view-model. Quite a lot of time was spent to style the areas and tool tips to look like the layout. And bind the hotspot colours to player colour palette exported from any layout based on our layout template.

With our initial settings qTip's auto positioning the tooltip did not work as well as in the test for the evaluation. I had to prefer tooltips to show on top and to the left of the hot spots. QTip should be able to find a good alternative position if the preferred did not have space for it. Problem was that qTip did not find the best alternative position for the tooltip. It positioned some of the tooltips partly outside the viewport even if there was plenty of room in another direction. These positioning problems were due to wrong settings for qTip. There are so many, and very advanced, settings in qTip that it can be very difficult to find the combination that works the best for each use case.

Another issue with tooltips was with the image in them. The qTip requires a third party jQuery plugin called 'imagesLoaded' to wait for images to load and only then position the tooltip. Without this plugin qTip uses the tooltip size without the image to position it. The issue with images was a problem because the sliding transition was planned to be used to show the tooltips. In the beginning of the transition tooltip's height is 0 pixels and then increases to its full height. The images load to the tooltips in the beginning of the transition, when the height is very small. Therefore the tooltips repositions wrong with images and sliding transition. This problem was solved by changing the show transition to a fade.

There was also an issue with iPad and possibly other touch devices as well. It was difficult to find the correct combination of show and hide events to also support tap events as fall back when using mouse over in non-touch devices. The result was that it needed an unfocus event in addition to mouseleave event to hide when tapped elsewhere for hiding the tooltips on iPad. Also it required click event in addition to mouseenter to show the tooltip. This worked on the first tap on the hotspot, but after hiding the tooltip it did not appear after the second tap on the hotspot. I did not investigate this further, but I think it had to do with the unfocus event in hide.

To make hotspots work better in mobile, instead of using jQuery's native events, HammerJS could be used. We have had good experiences using it instead of jQuery for event handling.

5.2 Carousel

I found 14 libraries that could somewhat compete for being selected for GHP, but it was too many for the final evaluation. I needed to go through them once more and screen out the top three or four. The basis for elimination was not meeting the following requirements:

- responsive design
- favourable licensing and pricing
- appropriate browser support
- touch support
- support for vertical mode

Some of the top libraries eliminated, not fulfilling these requirements, were listed in Subsection 4.3, Carousel.

The requirement for the vertical mode was too much even for the libraries with top features and quality. Only two libraries of the remaining passed this qualification. These were CarouFredSel and Swiper. They were the two libraries evaluated. That data for both libraries was found on their marketing, GitHub and technical documentation page, and my tests conducted in JSFiddle. Also CarouFredSel's demo gallery was also used to acquire information for the evaluation. All evaluation scores are in Table 3. I divided the original table into two parts, because the legend is very large as well. The legend gives insight on how

some of the properties were scored. The evaluation table shows good scores in green, average in yellow and bad ones in red.

Table 3. Swiper and CarouFredSel carousel library evaluation table

Requirement	Weight	Swiper [99] [100]		CarouFredSel [101] [102] [103]	
			Info		Info
Visuals					
Informative slide index	3	5	pagination with numbers	4	from API and demos
Colourable	4	5	tested	5	tested
Appearance very customizable	4	4	tested	4	tested
Various shape types	3	4	3 + 1 for custom support	3	non-native but adaptable from examples
Vertical carousel	5	4	tested	4	From API and a very simple example
Slides					
Scales by content when wanted	3	5	tested	2	API
HTML formatted text	5	5	tested	5	From examples
Function					
Supports 3D carousel	5	4	cover flow closest native shape to 3D carousel	2	seemed doable based on preliminary tests
Can show 2D carousel in phones	5	4	tested	4	API and examples
Pleasant animations	3	3	Demos and tests	3	from examples and test
User controls	5	5		4	API and examples
Accessible and intuitive navigation and controls	5	5		4	Examples
Touch support	5	5		3	API
Hint about gestures (animation)	2	2	API	2	API
Other					
Small file size	3	3	66 KB	4	54 KB
Good documentation	3	5		5	
Properly tested	2	4		4	
Actively developed/supported	4	5	March 27, 2015	1	April 16, 2013
Does not require other new libraries	3	5	Documentation	4	API
Free and friendly licence	4	5	MIT	5	MIT and GPL
Max (x5)	380				
SCORE		338		277	

As Table 3 could not fit a legend on the same page, it is separated as its own table, Table 1. It is not a typical legend. Table 4 provides information on how the properties meeting the requirements were rated as scores.

Table 4. Swiper and CarouFredSel carousel library evaluation table legend

Requirement	Legend Value / colour	5	4	3	2	1	0
	As percent	100	80	60	40	20	0
Normal requirement	Verbally	Perfectly, entirely, very much so	Yes, but not entirely, pretty much so	to some extent	some, a little, not much	very little	Not at all
Visuals							
Informative slide index		pagination and numbered index	other native and other doable	pagination or numbered index	no native, but doable	no native, but doable (tricky)	not doable
Various shape types	plus 1 if custom shapes enabled	> 5	5	4-3	2	1	0
Function							
Supports 3D carousel					doable		
Can show 2D carousel in phones		native option	as customization				
Touch support				needs a plugin			
Hint about gestures (animation)		very good native	ok native		no native, but easily doable	no native, but doable (tricky)	not doable
Other							
Small file size	KB, including css	< 30	< 60	< 100	< 200	< 300	> 300
Good documentation		Comprehensive API docs and good examples of different features	Comprehensive API docs and some examples of different features	ok docs and some examples	little docs and one example	little docs or one example	No docs no examples
Actively developed/supported	updated in (months)	3	6	12	20	30	> 30
Does not require other new libraries		yes	For advanced features	1	2	3	> 3
Free and friendly licence	Price (€)	0	< 10	< 30	< 60	< 100	> 100 or if licence cannot be bought just once

As Table 3 shows, Swiper got 338 points and CarouFredSel 277. Swiper's score is 89 % of the maximum and CarouFredSel's score 73 %. Swiper score is 22 % better than CarouFredSel's. Swiper got good or excellent grades for all requirements, but animation quality, hinting about gestures, and file size. File size was the only sector where CarouFredSel beat Swiper. CarouFredSel did poorly, including the same ones as Swiper, on: various carousel shape types, scaling by the content, touch support, and date last updated. Swiper does not actually support ring type 3D carousel as default, so I needed to make an advanced test to try if it is possible. It was pretty hard, but Swiper had some features that enable creating custom transitions. The overall excellent score for Swiper promised very successful implementation to the player.

Next was to implement the carousel to player as a new standardised activity, and to certify the proof of concept. Before I was able to do that, I had to implement it as a custom activity because of a tight schedule. Doing the custom implementation, I had the most difficulties loading the Swiper library to the player after jQuery was ready. Therefore the typical linking of JS libraries in the head element would not work, because jQuery was loaded with RequireJS afterwards. Also I could not use RequireJS, because it is only used by the libraries utilised in the player core. Custom implementations are not part of the core. I ended up us-

ing jQuery to successfully load Swiper library. The visual part of the customisation was fast and easy. I suppose this could have been just enough proof to certify the result of the evaluation. But the final implementation of the carousel features as a standardized activity was still required. It was very fluent and relatively quick. I decided to implement even more Swiper features I had originally planned, because it was so easy. The settings are managed from our course admin site. The name of the activity was changed to slide show to better describe the new broader range of features, including a simple image slide show with crossfade transition.

We have not had any client experiences from the standardized Carousel activity with Swiper. So there could be surprises, but I'm not expecting them. Personally the task for carousel tool was probably the most rewarding of the tasks in the study, as the library meets our requirements so well.

5.3 General Animation

For evaluating the general SVG animation libraries, I did not need to do any test on my own. I found a few good existing comparisons and necessary tests made with Codepen and jsFiddle where everyone can acknowledge the result.

The evaluation scores for general animation tools are presented in Table 5. The scoring uses the same 0 to 5 scale as all the other evaluations.

Table 5. Evaluation results for general SVG animation libraries.

	Requirement	Weight	GSAP	Velocity	PopMotion
Function	Timeline	4	5	3	0
	onUpdate call back	2	5	5	5
	onEnd call back	3	5	5	5
	Path and shape morphing	4	5	0	3
	Transform origin	4	5	0	0
	Performance	3	4	4	3
	Animation controllable with methods	4	5	2	4
Other	Works in IE9+	4	5	5	5
	Learning curve	3	3	3	3
	Small file size	4	2	4	4
	Good documentation	5	5	4	4
	Good examples	4	5	4	4
	Active community	3	5	4	3
	Score		214 / 235	150 / 235	152 / 235

Table 5 does not include the information on how the properties and features meeting the requirements were rated as scores. The following table, Table 6, shows that information.

Table 6. Evaluation table legend for general SVG animation libraries.

Legend						
Value / colour	5	4	3	2	1	0
As percent	100	80	60	40	20	0
Path and shape morphing	Both		Only one			None
Small file size	<20 KB	<40 KB	<70 KB	<110 KB	<200 KB	>200 KB
Others	Perfectly, entirely, very much so	Yes, but not entirely, pretty much so	to some extent	some, a little, not much	very little	Not at all

The evaluated timeline feature, in Table 5, does not refer to a visual timeline editor found in most of the animation IDEs such as Adobe Edge Animate, but to a supported functionality similar to a timeline in a YouTube video where user can pause, seek, and play a sequenced animation. GSAP was the only library to perform well in this sector.

All libraries had extensive call backs and proper browser support. GSAP is only one to supports both shape morphing and morphing line along a path. These are done with plugins that come with the paid GreenSock Club membership.

Transform origin feature gives better control on animated elements. The CSS property 'transform-origin' has been implemented differently or not at all to major browsers [104]. GSAP is the only library to unify this behaviour.

The performance points were given based on the libraries' ability to produce smooth animation with great quantities of animated elements. The points are based on a performance tester in Codepen called 'Speed Test: VelocityJS, GSAP, jQuery, and Transit'. PopMotion was not part of that test, but the creator of the library has assumed himself that it will lose to GSAP on a stress test. That is why it got weaker points. The other libraries have sufficient support for animation controls, but Velocity has only poor.

Velocity and PopMotion have small file size and GSAP fairly big. None have any problems with browser support and they all have equally moderate learning curves. All of the libraries have good documentation and examples. The activeness of the communities seems to be related to the age of the libraries so GSAP has the greatest, PopMotion the least and Velocity something in between.

The result of the evaluation and comparison is that GSAP suite gets selected as the general SVG animation library with excellent points. It got low points only for file size. This is definitely the best choice according to the results. Velocity does not even try to be as extensive which resulted to getting 0 points on 2 sectors for missing some important SVG specific features. What surprised me was PopMotion doing this well as a new comer. It got a few points more than Velocity. It also misses some key features required for advanced SVG animations, timeline and transform origin.

As a conclusion from the results one could say that GSAP performed well on all areas. Velocity and PopMotion did not perform as well, but their advantage to GSAP is the small file size. If all the advanced features of GSAP are not required, why not use a lighter library such as Velocity and PopMotion. As studying them I also run into some features that were missing or were executed better in one or both of them. These did not affect the evaluation, because they were not requirements to evaluated tools.

For proof of concept, I decided there is no need to implement new tests. The reason is that new tests would not produce more information about the library – at least its suitability as a general animation library. The extensive live examples of each feature in GSAP documentation, and the fact that we have already been using GSAP successfully as our main animation library in GHP for 2 years, should be enough proof of it.

5.4 Chart Animation

In the chart tool evaluation the number of libraries and other tools was overwhelming. It took over 35 hours to exclude the candidates with selected criteria and make the final evaluation. It should be noted that the selected tool is not the most suitable for every use case. There can be better libraries for each specific chart type or use case. In this task many libraries performed better than the selected tool in when evaluating the overall performance and suitability. But all the other top libraries failed in one or more of the critical areas. They were excluded from the final evaluation by price, animation capabilities, or they being based on canvas for rendering graphics.

After the elimination by the animation capabilities was done, there was only one library left for the final evaluation, C3.js, later just C3. Therefore there was no need for scoring just one tool's properties. Evaluation phase, such as carried out for the other tasks, was unne-

essary. I did still evaluate how C3 performed on the evaluation requirements to be able to report about it.

C3 is a library that relies on the popular D3, but it is easier to learn and use. The fact that development and latest release of the library was done during less than a year in 2014 and 2015, makes me wonder if it is still actively developed. But there is recent activity in the GitHub repository, which lead me to believe it will still be developed.

The overall look of the charts is very simple. I would like them to look more imposing to make the information in them more engaging. I suppose the animation has to be the part that makes the impression then, when it comes to the charts. The quality of the graphic design is satisfactory though. I would have also desired some 3D render modes as they can make the charts more appealing too. Some of the other top 13 chart libraries did support 3D mode for a few chart types.

The visual appearance of the charts is moderately easy to customise. So are the different functionalities. The texts in the charts cannot be formatted. This is a weakness that I did not find in the paid tools that made my top 13. C3 is also missing built-in text wrapping. Some of the other chart libraries did support it.

C3 does not support any export features, but as I studied how they were implemented to a few other libraries, it seemed possible to custom implement the feature to C3 too. This would be done with the help of the canvas element.

Minified file size is 140 kB for JS and 2kB for CSS. Also 149 kB has to be added for required D3 library so it is 291 kB total. That is a lot of code for the use cases when only one simple chart needs to be drawn. The amount of code can partly be explained with the number of different chart types supported – the number is impressive 16.

C3 has a comprehensive documentation, better than most of the chart libraries I looked into. The examples found on the site are also good, but so are the ones for the other libraries too. The community around C3 is not as good as the ones around the more popular chart tools.

For proof of concept I decided to make line, bar and pie charts in MPS company brand look and feel, and try to animate them as much as possible.

I managed to animate almost all the elements I wanted to. But I did have some issues. For line chart the animation starts with the axes values and the ticks scaling to the used range. Then, the first point of the first line of the data appears at $y=0$ and starts to vertically move to its real y -value. Just after the first point started to animate, after a short delay, the next point of the first line appears at $y=0$ and starts to animate to its correct y -value. As new points of the line appear the line forms between the points. All the points appear in order with an animation. When a line is ready the next line starts animating. After the lines are ready, the grey grid lines for the y values start to fade in one by one. Finally the legend fades in and the chart is ready. The Final stage of the line chart animation can be seen in Figure 9. The original animation can be found in <http://jsfiddle.net/prewiseDevelopers/0avvwsL9/>.

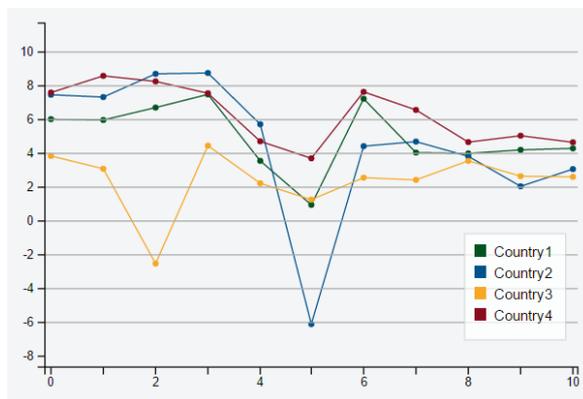


Figure 9. End state of line chart animation test with C3.js.

The column chart animates quite in the same manner as a line chart, but instead of points and lines, there are columns that rise upwards. I also had to remove animation from the grid lines. It is possible to set a separate x and y grids to make it easier to compare the columns heights to each other, with a setting in the initialization of a chart. This grid renders as dotted lines. To animate the grid to appear line by line, there is a method `chart.ygrids.add`, but it renders only solid lines which are rendered in front of the columns representing the data. The result looks heavy and decreases readability. So for the column chart, I felt it was necessary to remove the animated solid grid lines and just settle for non-animated dotted grid lines. The final stage of the column chart animation can be seen in Figure 10. URL to the original animation is <http://jsfiddle.net/prewiseDevelopers/0avvwsL9/>.

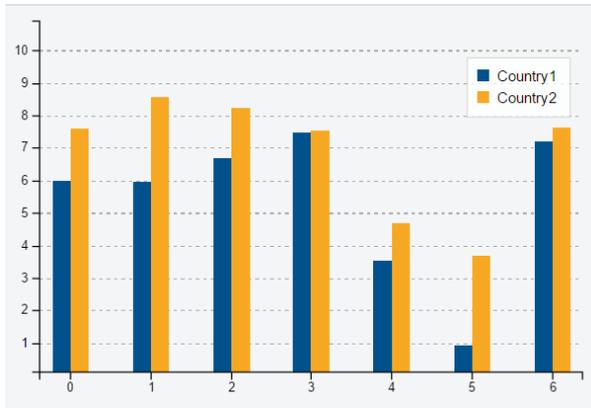


Figure 10. End state of column chart animation test with C3.js.

The pie chart has fewer elements than the two other tested chart types. I was able to animate all elements in the pie chart. The pie starts to animate slice by slice. At the start, the first slice fills the whole pie. The first slice is the green one, so at start, the whole pie becomes green. Then another slice appears as a white line, and starts to grow width taking area from the first slice. The growing stops when the relation between the two slice areas, is the same as it will be in the final state, with all the other slices added. The rest of the missing slices animate the same way, taking their own space in the pie. Last the legend appears. What I was missing, is a feature for animating the pie radius. I would like the pie to grow from small to large in the beginning of the animation. Also the texts, with the numeric values in them, seemed to flash annoyingly between the slice animations. The final stage of the pie chart animation can be seen in Figure 11. The original pie chart animation can be found in <http://jsfiddle.net/prewiseDevelopers/uoLfv79b/>.

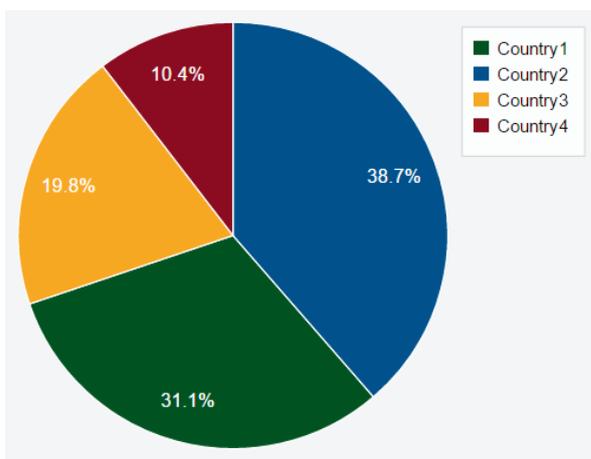


Figure 11. End state of pie chart animation test with C3.js.

All three animation tests conducted to C3 were successful. Almost every element of the charts could be animated. Not all elements can be animated as pleased though, but with some variation anyway. I accept this as proof of C3 being a good choice as an animated chart library.

As a conclusion, based on studying C3 and the other chart tools partly, I must say that C3 is not the most efficient chart library by far for general use. It does not even come close to having the most extensive features or the most appealing graphics. But it does have the most efficient animation capabilities of all chart libraries that otherwise would have suited to be used in GHP. If the weight had not been in the animation features, there would have been better candidates as a chart tool.

5.5 Character Animation

We were looking for tools to help managing the movement of the body parts and also a tool that would enable lip sync with audio. Because there were not any special tools for moving body parts, I suppose one could use any of the JavaScript libraries supporting SVG animation. I did some simple tests and the moving of the body parts could be done with just CSS and plain JavaScript. I used CSS Transitions to make the movement happen over a period of time. CSS Transitions do not work in IE9, but it was the easiest way to test the concept. The test can be found in <https://jsfiddle.net/prewiseDevelopers/c7z4g85u/>.

The most difficult and time consuming part was setting the points the body parts rotate around to – they are not the in the centre of the elements, nor in any of the “corners”. As default, SVG elements rotate around their centre point. Body parts should rotate around the point where they connect to other body parts – forearm rotates around the elbow and so on. I used SVG transform attribute with a translate value to set the correct rotation point for joints. This will not work in IE either, but was convenient for the test.

The test worked as planned. I managed to perform SVG character animations using forward kinematics utilised with CSS and some JS. At the time of writing this I am implementing a similar animation for a client utilising GSAP. After preliminary test, it seems GSAP handles the task very well, but as I said, I am just starting with this project. I can always run into problems later. At least the browser support should be better with GSAP. The tests for implementing lip sync are described next.

I managed to find one tool for creating lip sync, Lipsync by Sune Watts, but I could not get it working. It is basically a web site where user can upload an mp3. Then, as I understood, the site should interpret the speech in the file, and send an email with the timeline for phonemes or link where to download it. Instead of this, I got the server error 500, when trying to upload the mp3. This tool seemed to do what we needed on paper, so it was a pity it did not work at the time.

Because I could not find any, one, tool to help creating the lip sync, I tried to come up with a way to do it with several tools. The most difficult phase of making lip sync is interpreting the speech in the audio file, resulting to some kind of a timeline, with time codes for each phoneme or mouth movement. Of course this can be made manually, but it can save days of work having it done by a program. I knew this could be done in Flash, with an extension called SmartMouth. It was a welcomed surprise to notice it also had also an xml export. I created a pen SmartMouth XML to WebVTT in CodePen that converts the SmartMouth XML to WebVTT. It can be found in <http://codepen.io/makker/pen/zrLVJY>.

The converter output is then saved as a WebVTT file. Audio and WebVTT files were added to a page in a test course. We use Video.js as video and audio player. Now I needed to start listening 'cuechange' events to make a change in the mouth graphic according to the cue content. When I finally got that step working, I was not happy with the result. The animation was not natural. It seemed like a number of the mouth movements were not showing. I thought the problem was in SmartMouth. I did more research and found a software called Papagayo. It is also a tool for same purpose as SmartMouth, but it exports a .dat file that can be imported to professional 3D animation authoring tools Anime Studio or Blender. I made a similar converter as before to change the data to WebVTT format. It can be found in <http://codepen.io/makker/pen/EPdRbb?editors=0010>.

Making a lip sync map of phonemes with Papagayo is not as automated as in SmartMouth. First the transcript of the speech needs to be written or pasted on the audio file. Then the tool distributes the sentences on the audio timeline by their length so they are close to their correct places. Then the start and end of each sentence has to be shifted in sync with audio. The words are now distributed in the sentences, so they are also placed close to their correct places on the timeline, but they need to be fine-tuned manually. Phonemes are distributed with the words so now they also might require some fine-tuning too. This process is slow, but very fast compared to just doing all phoneme timing manually without any tools.

The end result was equally unsatisfying as it was with SmartMouth. It looked like the problem was not in the sync utilities. The problem had to be somewhere else. I made tests, where I put 24 cues per every second, and noticed only part of the cues triggered the 'cuechange' event in my tests. Only every 7th cue triggered a change event.

The result was better with setting fewer cues per second. I made another test tracing every time when Video.js 'timeupdate' event was released. The result was average of 3 times per second. I did not find the reason to this, but assumed the problem was probably in Video.js.

I made a new issue about my problem to Video.js GitHub site. I got an answer from another user, Gary Katsevman, telling that Video.js relies on browser's native 'timeupdate' event [105]. He further explained that the native timeupdate event's frequency depends on the browser and the conditions. If this is true, and I have to assume it is, it means that it is not possible to do the lip sync using the video element subtitles to handle the event triggers.

It seems my idea for implementing lip sync, did not work. It would be possible with even further research to study if there could still be an alternative implementation to make the synchronisation. Mozilla has made a library called Popcorn.js to sync media and other content. It could be possible to utilize it for the syncing. First, I would still try to achieve this with GSAP. It might work to set the mouth animation data to GreenSock's TimelineMax, and then try to keep audio and Timeline max in sync.

5.6 Discussion and Analysis

This subsection continues analysing the results, but on a more general level. It assesses how reliable the findings are and their significance to the company and to the field.

It also tries to put the findings to a perspective with the current knowledge in the field.

I feel I could not find enough academic sources for the study. There has not been enough study of the tools in this field. Therefore I had to rely heavily on the material I could find in the Web. There are quite a lot of blog posts, reviews, and different top something listings, but their reliability must be questioned. I must still say that they are often peer reviewed – in a way – in the comments below the main text. Also the significance of the writer to the field can be tested googling it and reviewing the results. I will not claim this fills the academic standards, but it was the only way to find the material for the evaluations.

I was probably stressing too much about how I could make sure that I would find all the suitable candidates for the evaluations as finding them was mainly based in making Google searches with key words. I did not want any good tools left out. I tried to adjust the queries to cover all aspects to the subject. But I could not come up with a system to be sure about it. That phase of the study could be conducted better, if conducted again. Also, I should have documented that phase better, so that it could be carried out by another party producing the same result. For the purpose the research was conducted, the accuracy of the results is adequate. I do not think there is any reason replicating it unless the requirements for the tools change.

I was surprised at how much information about the JS libraries was often not available on their web site, and I needed to make tests to gather all the information required for the evaluations. The requirements and the restrictions of responsive design and mobile support were not as significant a part of the study as I first assumed, and I felt they were often easy to fulfil. The library used for carousel had a very good support for RWD and mobile use. SVG seems to suite particularly well for RWD, based on the experiences with it.

I am not convinced it was worth the time using 50 hours searching, testing and evaluating the libraries for Hotspots. I could have just googled "best JS tooltip library", browsed the search results and I'm quite sure I would have picked qTip by gut feeling, saving 40 hours. The same waste of time happened partly with the general animation library and GSAP. I say partly, because finding a proper SVG animation tool with an editor would have been worth all the search hours. Finding a proper chart editor with efficient animation settings would have also saved a lot of hours when customising charts them to client projects. With carousel, I feel that the result was worth the time spent for the search and evaluation.

In general, I feel the overall results were partly surprising, partly expected. I expected that there would have been better suiting tools available. What I am most missing are the editors. General animations would greatly benefit from a proper editor.

I think the results of the activity tasks can be generalised to the field of e-learning and also to Web development. GHP added some special requirements, but I do not think they affected the result too much. The results of the animation tasks can be generalised to the field of SVG. For e-learning or web development not so much anymore, because including the canvas tools to evaluations would have most probably resulted to very different outcome.

I was very lucky to find informative articles discussing the usability issues and how to overcome them, when searching tools for carousel. The articles and studies rose very good points what to have as a requirement for the carousel tools. The Swiper library proved to suit well for the carousel implementation. We even added some of its alternative rendering types as options as one of the clients requested support for slide shows.

To take even more advantage of the carousel library, I had an idea to make a version of GHP with the page navigation and the appearance of a 3D carousel. In this implementation the page specific content areas, would be the slides of the carousel. This is definitely not suitable for smaller screen sizes, but could work in tablet size and bigger displays. The problem with really big screens, with width over 1800 pixels and 20 inches, is that we cannot scale the content area bigger than 1400 pixels wide – it just does not look well anymore. So there is a lot of empty space around the content area in large displays, and with 3D carousel we could use it to make the navigation transition more engaging.

As the study was nearly finished there appeared a need for a library for helping implement a 3 by 3 flip card grid. This would have suited well for the study, but the need for it appeared too late for adding it to the study. I assume a library could be found to help implementing the feature – a subject of future study.

Defining the animation categories was difficult, because there are no standards for it that would suit the purpose of this study. I also had a struggle selecting the made up categories that could have specialized libraries developed for them. The other categories of animation tools, excluded from this study, could be a subject for future study. My interest to canvas increased significantly towards to end of the research. Personally, that will be my next study subject.

I did find plenty of HTML5 animation editors, but they were only able to animate canvas or HTML5 elements, not elements of a SVG. Using SVG for Web animations still has issues. It is missing a proper editor for timeline animations. The support for SVG is different in the browsers. Standard is also missing some essential features, but they will surely be added in the forthcoming SVG 2.0.

The father of the Mobile First design concept, Luke Wroblewski, has said that SVG is a massive world of potential [106]. I have to agree. But I wish that the companies developing software for web development, such as Adobe, would also realize it, and make proper tools

to utilize that potential. Personally I'm also waiting for Dmitry Baranovskiy to return back to finalise the development of Snap.svg. I also wish Abode would start using it as the SVG manipulation library in Adobe Animate. It would be a perfect match.

I believe that Adobe Animate will become the industry standard for HTML5 animations, like many other Adobe products have become in their own fields. I also hope the support for SVG based animations will be developed to support embedded audio, texts and especially scripting to enable interactive features for the animation.

Before getting the study ready I got a client project where I need the use of inverse kinematics in a character animation. Because I was not able to find proper tools for implementing IK with SVG, I will have to use a different method. I will basically try to save the angle values of the joints in enough postures that I can use interpolation to calculate the missing postures between the pre-calculated states. I might also try to create a simple inverse kinematics library myself. I have not decided yet.

6 Summary

This chapter sums up the key issues covered in the study. The research question that the thesis tried to answer is: What are the best tools and technologies to develop standardized activities and custom SVG animations for e-learning with HTML5 technologies taking into account the requirements and the restrictions of the mobile use and devices?

The study tried to answer the question by dividing the studied topics into subcategories and selecting the tools and technologies best suited for each category and trying to get prove they are the best choices with tests.

The method to find the best tool was evaluating and scoring the tool candidates, and their properties, for each subcategory against each other. The data used in the research was collected from a great number of sources. Some of them were more reliable than others.

The tests, for getting proof that each selected tools suited well for their function, were implementing the activities to the developed software, Gimlet HTML5 Player, and making test animations for the animation subcategories. Table 7 presents the subcategories and the tools selected for them.

Table 7. The subcategories and the selected tools

Hotspots	None, but qTip is great help with tooltips
Carousel	Swiper
General Animation	GSAP
Chart animation	C3.js
Character	None, but could benefit from GSAP in implementation

As seen in Table 7, a proper tool was not found for every category. Swiper and GSAP can be well recommended for use cases in their categories. SVG animation tools are missing a proper editor. No specific tools could be found that would be targeted to or could be benefited from implementing SVG character animations. Surely character animations can be done with SVG graphics, but the animation needs to be produced with custom code.

All selected tools enable easy visual customization. They also support well the RWD design pattern. As Adobe develops further its Adobe Animate, and implement better support for SVG animations I believe it will be a great asset in all SVG animation.

References

- 1 MPS Prewrite Oy. Prewrite Client Course Statistics [Excel spreadsheet]. 2013. [Unpublished internal document] Accessed 6 November 2014.
- 2 Rosenblatt S. Why does your company force you to use IE? [Online]. Download.com; 12 February 2013.
URL: <http://download.cnet.com/blog/download-blog/why-does-your-company-force-you-to-use-ie>. Accessed 4 April 2015.
- 3 Nicholson P. A History of E-Learning. In Computers and Education. Burwood, Australia: Springer Netherlands; 2007. p. 1-11.
- 4 Deloitte Development LLC. Bersin - Lexicon. [Online]. no date.
URL: <http://www.bersin.com/Lexicon/>. Accessed 16 April 2014.
- 5 Toth TA. The Right e-Learning Tool for The Job. In Allen MW, editor. MICHAEL ALLEN'S 2012 e-LEARNING ANNUAL [Online]. San Francisco, CA: Pfeiffer; 2011.
URL: <http://tinyurl.com/gnml9d>. Accessed 23 April 2014.
- 6 Mallon D. m-Learning: Mobile Learning Is Finally Going Mainstream – And It Is Bigger Than You Might Think. Bersin & Associates; 2011.
- 7 Crompton H. A historical overview of mobile learning: Toward learner-centered education. In Z. L. Berge LYM, editor. Handbook of mobile learning.: Routledge; 2013. p. 3-14.
- 8 Hay S. Stephen Hay. [Online]. Twitter; 7 January 2011.
URL: <https://twitter.com/stephenhay/status/23350345962889216>. Accessed 8 April 2015.
- 9 Firtman M. Programming the Mobile Web. 2nd ed. Sebastopol: O'Reilly Media Inc.; 2010.
- 10 StatCounter. StatCounter Global Stats. [Online]. 2016.
URL: <http://gs.statcounter.com/>. Accessed 27 March 2016.
- 11 Ruottu M, et al.. Course Version Log [Excel spreadsheet]. MPS Prewrite; 2008. [Unpublished internal document] Accessed 6 November 2014.
- 12 Katz J. Learning Solutions Magazine. [Online]. The eLearning Guild; 31 May 2010.
URL: <http://tinyurl.com/ho9slsu>. Accessed 30 December 2014.
- 13 Shank , Ganci J. eLearning Authoring Tools 2013: What We're Using, What We Want. Santa Rosa, CA: The eLearning Guild; 2013.
- 14 Souders S. Internet Archive. [Online]. 2015.
URL: <http://www.httparhive.org/>. Accessed 20 March 2015.
- 15 Grover C. Adobe Edge Animate: The Missing Manual. Sebastopol, CA: O'Reilly Media; 2012.
- 16 Clark RC, Mayer RE. e-Learning Architectures. In e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning. 3rd ed. [Online].: Pfeiffer; 2011.
URL: <http://tinyurl.com/oo4nvwo>. Accessed 17 April 2014.

- 17 Google. Responsive interaction - Animation - Google design guidelines. [Online]. no date.
URL: <http://www.google.com/design/spec/animation/responsive-interaction.html>. Accessed 10 March 2015.
- 18 Griol D, Callejas Z, López-Cózar R. Technologies for Inclusive Education. IGI Global; 2012.
- 19 Clark RC, Mayer RE. Should You Change Static Illustrations into Animations? In e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning. 3rd ed. [Online].: Pfeiffer; 2011.
URL: <http://tinyurl.com/l4pntuu>. Accessed 19 April 2014.
- 20 Tversky B, et al. Enriching animations. In Lowe R, Schnotz W. Learning with Animation: Research Implications for Design.: Cambridge University Press; 2008. p. 263-285.
- 21 Mayer RE, Moreno R. Animation as an Aid to Multimedia Learning. In Educational Psychology Review Vol. 14, No. 1.: Kluwer Academic Publishers - Plenum Publishers; 2002. p. 87-99.
- 22 Nissinen H. (hanna.nissinen@prewise.fi) VS: Animaatioiden käytöstä. [Online]. E-mail to: Markus Ruottu (markus.ruottu@prewise.fi) 23 Apr 2014;
- 23 Jiménez-Castillo D, Sánchez Fernández R. The Impact of Combining Video Podcasting and Lectures on Students' Assimilation of Additional Knowledge: An Empirical Examination. In Pelet JE, editor. E-Learning 2.0 Technologies and Web Applications in Higher Education [Online]. Hershey PA: IGI Global; 2013.
URL: <http://tinyurl.com/jknqgsp>. Accessed 16 April 2014.
- 24 Clark RC, Mayer RE. Psychological Reasons for the Segmenting Principle. In e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning. 3rd ed. [Online].: Pfeiffer; 2011.
URL: <http://tinyurl.com/jhnvv7t>. Accessed 19 April 2014.
- 25 Clark RC, Ann K. Visuals and Learning. In The New Virtual Classroom: Evidence-Based Guidelines for Synchronous e-Learning [Online].: Pfeiffer; 2007.
URL: <http://tinyurl.com/j24keq8>. Accessed 4 May 2014.
- 26 Clark RC MR. Applying the Personalization Principle. In e-Learning Architectures. In e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning. 3rd ed. [Online].: Pfeiffer; 2011.
URL: <http://tinyurl.com/zowfz8a>. Accessed 27 April 2014.
- 27 Clark RC, Richard ME. Personalization Principle 2: Use Effective On-Screen Coaches to Promote Learning. In e-Learning and the Science of Instruction: Proven Guidelines for Consumers and Designers of Multimedia Learning. 3rd ed. [Online]. San Francisco, CA: Pfeiffer; 2011.
URL: <http://tinyurl.com/hz2flum>. Accessed 26 April 2014.
- 28 Leenheer N. How well does your browser support html5? [Online]. HTML5TEST; no date.
URL: <http://html5test.com/results/desktop.html>. Accessed 29 May 2014.
- 29 HTML5 PUBLICATION HISTORY - W3C. [Online]. W3C;
URL: <http://www.w3.org/standards/history/html5>. Accessed 12 April 2015.

- 30 Wilcox M. Responsive Images: What's the Problem, and How Do We Fix It? [Online]. 13 June 2012.
URL: <https://dev.opera.com/articles/responsive-images-problem/>. Accessed 17 March 2015.
- 31 Firdaus T. Responsive Web Design by Example. Birmingham: Packt Publishing Ltd.; 2013.
- 32 W3C. Visual formatting model. [Online]. 7 June 2011.
URL: <http://www.w3.org/TR/CSS21/visuren.html>. Accessed 16 March 2015.
- 33 Stow M. Viewport Sizes. [Online].
URL: <http://viewportsizes.com/?filter=iPhone>. Accessed 16 March 2015.
- 34 Apple Inc. Apple - iPhone 6 - Technical Specifications. [Online]. 2015.
URL: <https://www.apple.com/iphone-6/specs/>. Accessed 16 March 2015.
- 35 Inc. G. Web Fundamentals — Set the viewport. [Online]. 2014.
URL: <http://tinyurl.com/n3ecqu2>. Accessed 16 March 2015.
- 36 Marcotte E. Responsive Web Design. New York: A Book Apart; 2011.
- 37 McFarland DS. Dreamweaver CS6: The Missing Manual. Sebastopol: O'Reilly Media, Inc.; 2012.
- 38 Parhi P, Karlson AK, Bederson BB. Target Size Study for One-Handed Thumb Use on Small Touchscreen Devices. In MobileHCI '06 Proceedings of the 8th conference on Human-computer interaction with mobile devices and services; 2006; Espoo. p. 203-210.
- 39 Wroblewski L. Multi-Device Web Design. [Online].; 2012 Accessed 3 March 2015.
Available from: http://static.lukew.com/onedesign_04092013.pdf.
- 40 Deveria A. Can I use. [Online]. no date.
URL: <http://caniuse.com/>. Accessed 2 March 2015.
- 41 Lamba J. Food Sense. [Online]. 2015.
URL: <http://foodsense.is/>. Accessed 17 March 2015.
- 42 Harrison J. Responsive Icons. [Online].
URL: <http://www.joeharrison.co.uk/projects/responsiveicons>. Accessed 22 April 2015.
- 43 Dailey D, Frost J, Strazzullo D. Building Web Applications with SVG. Sebastopol, California: O'Reilly Media, Inc.; 2012.
- 44 Čereška S. (Simon.Cereska@prewise.com) Re: Questions for my thesis. [Online].
Email to: Markus Ruottu (markus.ruottu@prewise.fi); 8 April 2016.
- 45 Osmani A. Learning JavaScript Design Patterns. 1st ed. O'Reilly Media; 2012.
- 46 Barnard EM. KnockoutJS Starter. Birmingham: Packt Publishing; 2012.
- 47 Kumar V. HTML5 and JavaScript Apps with MVVM and Knockout. In Beginning Windows 8 Data Development. NY, US: Apress; 2013. p. 13-27.
- 48 knockoutjs.com. Knockout. [Online]. no date.
URL: <http://knockoutjs.com/>. Accessed 9 March 2015.

- 49 Osmani A. Journey Through The JavaScript MVC Jungle. [Online]. Smashing Magazine; 27 July 2012.
URL: <http://www.smashingmagazine.com/2012/07/27/journey-through-the-javascript-mvc-jungle/>. Accessed 9 March 2015.
- 50 Osmani A. Writing Modular JavaScript With AMD, CommonJS & ES Harmony. [Online]. no date.
URL: <http://addyosmani.com/writing-modular-js/>. Accessed 26 December 2014.
- 51 Wikimedia Foundation Inc. Asynchronous module definition. [Online]. Wikipedia; 2014.
URL: http://en.wikipedia.org/wiki/Asynchronous_module_definition. Accessed 29 December 2014.
- 52 Burke J. Require.JS - Why AMD? [Online]. no date.
URL: <http://requirejs.org/docs/whyamd.html>. Accessed 29 December 2014.
- 53 Khalsa MSSS. Introduction to AMD Modules. [Online]. Sitepen; no date.
URL: <http://dojotoolkit.org/documentation/tutorials/1.9/modules/>. Accessed 29 December 2014.
- 54 Amdjs. GitHub - AMD. [Online]. 2014.
URL: <https://github.com/amdjs/amdjs-api/wiki/AMD>. Accessed 23 December 2014.
- 55 Nimesh R. SitePoint. [Online]. 7 January 2013.
URL: <http://www.sitepoint.com/understanding-requirejs-for-effective-javascript-module-loading/>. Accessed 22 December 2014.
- 56 Odell D. Alternatives To RequireJS. In Pro JavaScript Development: Coding, Capabilities, and Tooling [Online].: Apress; 2014.
URL: <http://tinyurl.com/jgusnw9>. Accessed 22 December 2014.
- 57 Tuomala E. (<https://www.facebook.com/hopeatussi>) [Facebook] Private message to Markus Ruottu. [Online]. 9 April 2016.
- 58 Cecco R. Supercharged JavaScript Graphics. Sebastopol, CA: O'Reilly Media, Inc.; 2011.
- 59 Pyrhönen P. (pekka.pyrhonen@prewise.fi) VS: Kyssäri. [Online]. Email to: Markus Ruottu (markus.ruottu@prewise.fi); 8 April 2016.
- 60 W3C. Standards for Web Applications on Mobile: current state and roadmap. [Online]. 2015.
URL: <http://www.w3.org/Mobile/mobile-web-app-state/>. Accessed 16 March 2015.
- 61 SVG Full 1.1 2nd Edition - Conformance Suite Implementation Status. [Online]. W3C SVG Working Group; 9 May 2011.
URL: http://dev.w3.org/SVG/profiles/1.1F2/test/status/implementation_matrix.html. Accessed 6 January 2016.
- 62 Murray S. Interactive Data Visualization for the Web. Sebastopol, CA: O'Reilly Media, Inc.; 2013.
- 63 W3C. SVG 1.1 (Second Edition) - Text. [Online]. 16 August 2011.
URL: <http://www.w3.org/TR/SVG11/text.html>. Accessed 25 February 2015.
- 64 HTML5 Canvas. 2nd ed. Sebastopol, CA: O'Reilly Media, Inc.; 2013.
- 65 Soueidan S. CSS vs SVG: The Final Round(up) : Adobe Dreamweaver Team Blog.

- [Online]. Adobe Systems Incorporated; 16 September 2015.
URL: <http://blogs.adobe.com/dreamweaver/2015/09/css-vs-svg-the-final-roundup.html>.
Accessed 11 February 2016.
- 66 Nabors R. The State Of Animation 2014. [Online]. Smashing Magazine; 18 November 2014.
URL: <http://www.smashingmagazine.com/2014/11/18/the-state-of-animation-2014/>.
Accessed 10 March 2015.
- 67 Deveria A. Can I Use - Web Animations API. [Online]. no date.
URL: <http://caniuse.com/#feat=web-animation>. Accessed 10 May 2015.
- 68 Ganci J. Seven Top Authoring Tools. [Online]. Magazine, Learning Solutions; 10 December 2011.
URL: <http://www.learningsolutionsmag.com/articles/768/seven-top-authoring-tools>.
Accessed 26 March 2016.
- 69 Pernice K. Designing Effective Carousels: Create a Fanciful Amusement, Not a House of Horrors. [Online]. Nielsen Norman Group; 14 September 2013.
URL: <http://www.nngroup.com/articles/designing-effective-carousels/>. Accessed 24 March 2015.
- 70 Future Web Design. Ultimate 3D Carousel. [Online].
URL: <http://www.webdesign-flash.ro/p/u3dcar/>. Accessed 24 March 2015.
- 71 Peatt K. An Exploration Of Carousel Usage On Mobile E-Commerce Websites. [Online]. Smashing Magazine; 9 February 2015.
URL: <http://tinyurl.com/nxpy7jb>. Accessed 23 March 2015.
- 72 Nielsen J. Auto-Forwarding Carousels and Accordions Annoy Users and Reduce Visibility. [Online]. Nielsen Norman Group; 19 January 2013.
URL: <http://www.nngroup.com/articles/auto-forwarding/>. Accessed 23 March 2015.
- 73 Frost B. Carousels. [Online]. 23 January 2013.
URL: <http://bradfrost.com/blog/post/carousels/>. Accessed 23 March 2015.
- 74 Runyon E. Carousel Interaction Stats. [Online]. WeedyGarden; 22 January 2013.
URL: <http://erikrunyon.com/2013/01/carousel-stats/>. Accessed 23 March 2015.
- 75 Design FW. Royal 3D Carousel. [Online].
URL: <http://www.webdesign-flash.ro/p/royal-3d-carousel/>. Accessed 24 March 2015.
- 76 Brown S. jQuery Circular Carousel. [Online]. GitHub; 24 July 2014.
URL: <https://github.com/samuelgbrown/jquery.circular-carousel>. Accessed 27 March 2015.
- 77 CrayThemes. Storyline 3D Slider. [Online]. Envato Pty Ltd.; 27 October 2013.
URL: <http://codecanyon.net/item/storyline-3d-slider/5951671>. Accessed 26 March 2015.
- 78 Gechev M. jQCarousel - jQuery UI Plugin. [Online].
URL: <http://carousel.mgechev.com/>. Accessed 27 March 2015.
- 79 Adobe Systems Software Ireland Ltd.. Adobe.com. [Online].
URL: <https://creative.adobe.com/addons/products/12329#.VtHJCZyLT9Y>. Accessed 27 February 2016.

- 80 Adobe Systems Incorporated. Update about Edge Tools and Services. [Online]. Creative Cloud blog by Adobe; 30 November 2015.
URL: <http://blogs.adobe.com/creativecloud/update-about-edge-tools-and-services/>. Accessed 27 February 2016.
- 81 Tudor A. Transforms on SVG Elements. [Online]. CSS-Tricks; 5 March 2015.
URL: <https://css-tricks.com/transforms-on-svg-elements/>. Accessed 1 March 2016.
- 82 Janaud S. JavaScript Chart library - JenScript JS. [Online].
URL: <http://www.jenscript.io/>. Accessed 14 February 2016.
- 83 Shevchuk A. Online Charts Builder. [Online]. 2015.
URL: <http://charts.hohli.com/new/index.html>. Accessed 14 February 2016.
- 84 Shutterstock, Inc. Rickshaw Examples. [Online]. 2013.
URL: <http://code.shutterstock.com/rickshaw/examples/>. Accessed 9 February 2016.
- 85 tenXer, Inc. xCharts » Documentation. [Online]. 2012.
URL: <http://tenxer.github.io/xcharts/docs/#options>. Accessed 9 February 2016.
- 86 DensityDesign. Raw. [Online]. Design Department of the Politecnico di Milano;
URL: <http://raw.densitydesign.org/>. Accessed 12 February 2016.
- 87 Stanford Visualization Group. Protovis. [Online]. 2010.
URL: <http://mbostock.github.io/protovis/>. Accessed 9 February 2016.
- 88 The Next Web, Inc. 20 best JavaScript charting libraries. [Online]. 12 June 2015.
URL: <http://thenextweb.com/dd/2015/06/12/20-best-javascript-chart-libraries/#gref>. Accessed 9 February 2016.
- 89 Sencha Inc.. Ext JS - JavaScript MVC/MVVM framework for cross-platform web apps. [Online]. 2016.
URL: <https://www.sencha.com/products/extjs/#overview>. Accessed 8 February 2016.
- 90 Kunz G. Chartist - Simple responsive charts. [Online].
URL: <http://gionkunz.github.io/chartist-js/index.html>. Accessed 8 February 2016.
- 91 novus/nvd3: A reusable charting library written in d3.js. [Online]. 24 January 2016.
URL: <https://github.com/novus/nvd3>. Accessed 9 February 2016.
- 92 TechSlides. 50 JavaScript Libraries for Charts and Graphs | TechSlides. [Online]. 1 August 2015.
URL: <http://techslides.com/50-javascript-charting-and-graphics-libraries>. Accessed 10 February 2016.
- 93 InfoSoft Global Private Limited. JavaScript Charts for Web and Mobile - FusionCharts. [Online]. 2016.
URL: <http://www.fusioncharts.com/>. Accessed 18 February 2016.
- 94 Ajar Productions. SmartMouth™ Help Documentation. [Online]. 2014.
URL: <http://ajarproductions.com/pages/products/smartmouth/help.php>. Accessed 26 April 2014.
- 95 CreateJS | Demos for EaselJS - Sprite Sheets. [Online]. gskinner;
URL: <http://www.createjs.com/Demos/EaselJS/SpriteSheet>. Accessed 20 May 2015.
- 96 CrazyTalk Animator 2 Features. [Online]. Reallusion Inc.;
URL: http://www.reallusion.com/crazytalk/Animator/animator_feature_animation.aspx.

Accessed 20 May 2015.

- 97 Envato Pty Ltd. Hotspot Map - Powerful annotations and tooltips. [Online]. 2014.
URL: <http://tinyurl.com/h4e3y2k>. Accessed 7 November 2014.
- 98 inabrains.com. LiteTooltip.js Bundle 2.0. [Online]. 2013.
URL: <http://www.inabrains.com/tooltip/index.html>. Accessed 7 November 2014.
- 99 nolimits4web. Swiper. [Online]. GitHub;
URL: <https://github.com/nolimits4web/Swiper>. Accessed 25 March 2015.
- 100 nolimits4web. Swiper - Most Modern Mobile Touch Slider. [Online]. iDangero.us;
URL: <http://www.idangero.us/swiper/>. Accessed 26 March 2015.
- 101 Dev7studios. carouFredSel - Dev7studios Docs. [Online].
URL: <http://docs.dev7studios.com/jquery-plugins/caroufredsel>. Accessed 31 March 2015.
- 102 Pellegroni G. carouFredSel. [Online]. GitHub, Inc.;
URL: <https://github.com/gilbitron/carouFredSel>. Accessed 31 March 2015.
- 103 Fred. CoolCarousels - Showcasing 72 cool jQuery carousel examples and tutorials. [Online]. Fredsite.nl;
URL: <http://coolcarousels.fredsite.nl/>. Accessed 31 March 2015.
- 104 Doyle J. SVG Animation and CSS Transforms: A Complicated Love Story. [Online]. CSS Tricks; 10 November 2014.
URL: <https://css-tricks.com/svg-animation-on-css-transforms/>. Accessed 2 March 2016.
- 105 Katsevman G. Issuing cues on TextTrack · Issue #3089 · videojs/video.js. [Online]. GitHub; 8 February 2016.
URL: <https://github.com/videojs/video.js/issues/3089>. Accessed 30 March 2016.
- 106 Wroblewski L. An Event Apart: SVG is for Everybody. [Online]. 23 September 2014.
URL: <http://www.lukew.com/ff/entry.asp?1921>. Accessed 17 March 2015.
- 107 Raaska T. Prewise Gimlet LMS HTML5 support [Word document]. MPS Prewise; 2014. [Unpublished internal document] Accessed 6 November 2014.
- 108 Brightcove Inc. HTML5 Video Player | Video.js. [Online]. 2014.
URL: <http://www.videojs.com/>. Accessed 25 March 2014.
- 109 Brightcove Inc. Github - videojs. [Online]. no date.
URL: <https://github.com/videojs/video.js/wiki>. Accessed 26 March 2014.
- 110 Wroblewski L. LukeW Ideation & Design. [Online]. no date.
URL: <http://www.lukew.com/>. Accessed 19 November 2013.
- 111 Wang H. Interactivity in E-Learning [Online]. Hershey, PA: IGI Global; 2011.
URL: <http://my.safaribooksonline.com/book/-/9781613504413>. Accessed 16 April 2014.
- 112 Schone BJ. Engaging Interactions for eLearning - 25 Ways to Keep Learners Awake and Intrigued. Initial release ed. [Online]. eLearningWeekly.com; 2007.
URL:
<http://online.fiu.edu/files/newsletter/issue07/EngagingInteractionsForELearning.pdf>.
Accessed 5.11.2014 November 2014.

- 113 MPS Prewrite Oy. Company - Prewrite. [Online]. 2014.
URL: <http://www.prewrite.com/en/company/about>. Accessed 18 April 2014.
- 114 Mozilla. GitHub - Shumway. [Online]. 2014.
URL: <https://github.com/mozilla/shumway>. Accessed 19 December 2014.
- 115 Cladera. GitHub - VideoJS Cuepoints. [Online]. 2014.
URL: <https://github.com/cladera/videojs-cuepoints>. Accessed 26 March 2014.
- 116 Arriaga AF. JavaScript Web Applications - Building web applications using Asynchronous Module (AMD). Bachelor's Thesis. Helsinki Metropolia University of Applied Sciences; 2013.
- 117 Adobe Web Platform Team. Snap.svg. [Online]. no date.
URL: <http://snapsvg.io/about/>. Accessed 2 March 2015.
- 118 W3C. Mobile Web Application Best Practices. [Online]. 14 December 2010.
URL: <http://www.w3.org/TR/mwabp/>. Accessed 16 March 2015.
- 119 Wroblewski L. On Using Browser Viewports to Understand Screens. [Online]. 2 October 2013.
URL: <http://www.lukew.com/ff/entry.asp?1817>. Accessed 16 March 2015.
- 120 DeSandro D. Creating responsive, touch-friendly carousels with Flickity. [Online]. CSS-Tricks; 5 March 2015.
URL: <https://css-tricks.com/creating-responsive-touch-friendly-carousels-with-flickity/>. Accessed 23 March 2015.
- 121 Gechev M. jqcarousel. [Online]. GitHub;
URL: <https://github.com/mgechev/jqcarousel>. Accessed 26 March 2015.
- 122 Freb. Frebsite. [Online].
URL: <http://www.frebsite.nl/>. Accessed 31 March 2015.
- 123 Coyier C. Sass vs. LESS - CSS-Tricks. [Online]. 16 May 2012.
URL: <https://css-tricks.com/sass-vs-less/>. Accessed 25 April 2015.
- 124 Catlin H, Weizenbaum N, Eppstein C. Sass: Sass Basics. [Online].
URL: <http://sass-lang.com/guide>. Accessed 25 April 2015.
- 125 Watts S. Lipsync. [Online]. 2016.
URL: <http://sunewatts.dk/lipsync/>. Accessed 2 February 2016.
- 126 Lost Marble. Papagayo. [Online]. 2016.
URL: <http://lostmarble.com/papagayo-home/>. Accessed 7 February 2016.
- 127 The Mozilla Foundation. Popcorn.js | The HTML5 Media Framework. [Online].
URL: <http://popcornjs.org/>. Accessed 7 February 2016.
- 128 thisgeek. [Online].
URL: <http://jsfiddle.net/thisgeek/At3xg/>. Accessed 7 February 2016.
- 129 Envato. 10 Open-Source JavaScript Data Chart Libraries Worth Considering - Blog. [Online]. Melbourne: 31 March 2015.
URL: <http://marketblog.envato.com/resources/open-source-javascript-data-chart-libraries/>. Accessed 9 February 2016.
- 130 ZingChart. Commercial JavaScript Charts - Licensing Options | ZingChart. [Online].

2016.
URL: <https://www.zingchart.com/buy/>. Accessed 9 February 2016.
- 131 Zygomatic. Online Charts. [Online].
URL: <http://www.onlinecharttool.com/>. Accessed 11 February 2016.
- 132 ChartBlocks. Online Chart Builder - ChartBlocks. [Online]. 2014.
URL: <http://www.chartblocks.com/en/>. Accessed 11 February 2016.
- 133 SocialCompare. Javascript Graphs and Charts libraries | Comparison tables. [Online]. 7 February 2016.
URL: <http://socialcompare.com/en/comparison/javascript-graphs-and-charts-libraries>. Accessed 13 February 2016.
- 134 Vaadin. Charts for JavaScript - vaadin.com. [Online].
URL: <https://vaadin.com/charts-for-javascript/>. Accessed 13 February 2016.
- 135 Bostock M. D3.js - Data-Driven Documents. [Online]. 2015.
URL: <http://D3js.org>. Accessed 18 February 2016.
- 136 Bostock M. mbostock/d3: A JavaScript visualization library for HTML and SVG. [Online]. 2015.
URL: <https://github.com/mbostock/d3>. Accessed 18 February 2016.
- 137 JSXGraph team. JSXGraph - JSXGraph. [Online]. Bayreuth: Center for Mobile Learning with Digital Technology – Universität Bayreuth; 2016.
URL: <http://jsxgraph.uni-bayreuth.de/wp/>. Accessed 18 February 2016.
- 138 AnyChart.Com. Interactive cross-platform JavaScript HTML5 Charts, Maps, Stocks and Gantts for your project | AnyChart. [Online]. 2016.
URL: <http://www.anychart.com/>. Accessed 18 February 2016.
- 139 Developer Express Inc. Charts and Data Visualization HTML5/JS Desktop Widgets. [Online].
URL: <http://js.devexpress.com/WebDevelopment/Charts/>. Accessed 18 February 2016.
- 140 Shield UI Ltd. Chart | Shield UI. [Online].
URL: <http://www.shieldui.com/products/chart>. Accessed 18 February 2016.
- 141 Palantir Technologies. Palantir - Plottable.js. [Online].
URL: <http://plottablejs.org/components/>. Accessed 18 February 2016.
- 142 Levine S. dc.js - Dimensional Charting Javascript Library. [Online].
URL: <http://dc-js.github.io/dc.js/>. Accessed 18 February 2016.
- 143 Kiernander J. dimple - A simple charting API for d3 data visualisations. [Online].
URL: <http://dimplejs.org/>. Accessed 18 February 2016.
- 144 Simoes A. D3plus. [Online].
URL: <http://d3plus.org/>. Accessed 18 February 2016.
- 145 Amcharts.com. JavaScript Charts | amCharts. [Online].
URL: <https://www.amcharts.com/javascript-charts/>. Accessed 21 February 2016.
- 146 Tanaka M. C3.js | D3-based reusable chart library. [Online]. 2014.
URL: <http://c3js.org/>. Accessed 21 February 2016.
- 147 Tanaka M. masayuki0812/c3: A D3-based reusable chart library. [Online].

- URL: <https://github.com/masayuki0812/c3>. Accessed 21 February 2016.
- 148 Animatron. HTML5 Online Animation Editor | Animatron. [Online]. Boston: 2016.
URL: <https://www.animatron.com/>. Accessed 25 February 2016.
- 149 2016 Adobe Systems Incorporated. HTML5, vector animation software | Adobe Animate CC (Flash Pro). [Online]. 2016.
URL: <http://www.adobe.com/products/animate.html>. Accessed 27 February 2016.
- 150 Animachine. Animachine. [Online]. GitHub, Inc.;
URL: <https://github.com/animachine/animachine>. Accessed 28 February 2016.
- 151 Orrù F. Tweene : JavaScript Animation Proxy. [Online].
URL: <http://tweene.com/>. Accessed 28 February 2016.
- 152 Quasso. Quasso/qui: A UI pack of common transitions for front-end devs that uses GSAP as the animation engine. [Online]. GitHub, Inc.;
URL: <https://github.com/Quasso/qui>. Accessed 28 February 2016.
- 153 GreenSock. Speed Test: VelocityJS, GSAP, jQuery, and Transit. [Online].
URL: <http://codepen.io/GreenSock/pen/pmknl>. Accessed 2 March 2016.
- 154 GreenSock. Speed Test: VelocityJS, GSAP, jQuery, and Transit. [Online]. Codepen;
URL: <http://codepen.io/GreenSock/pen/pmknl>. Accessed 2 March 2016.
- 155 Shapiro J. Velocity.js. [Online].
URL: <http://julian.com/research/velocity/>. Accessed 2 March 2016.
- 156 MPS Prewise Oy. Prewise Gimlet Learning Suite - Prewise. [Online]. 2016.
URL: <http://www.mps.fi/en/mps-prewise/technology/prewise-gimlet-learning-suite.html>.
Accessed 2 March 2016.
- 157 Khronos Group. Getting Started - WebGL Public Wiki. [Online]. 10 April 2011.
URL: https://www.khronos.org/webgl/wiki/Getting_Started. Accessed 26 March 2016.
- 158 Drasner S. Weighing SVG Animation Techniques (with Benchmarks). [Online]. CSS-Tricks; 27 January 2015.
URL: <https://css-tricks.com/weighing-svg-animation-techniques-benchmarks/>.
Accessed 29 March 2016.
- 159 Clark RC. Developing Technical Training: A Structured Approach for Developing Classroom and Computer-based Instructional Materials. 3rd ed. [Online]. San Francisco, CA: Pfeiffer; 2007.
URL: <http://tinyurl.com/hbxswcp>. Accessed 11 May 2004.
- 160 Perez-Gonzalez D, Soto-Acosta P, Popa S. Effective Implementation of an Interuniversity E-Learning Initiative. In Pelet JE, editor. E-Learning 2.0 Technologies and Web Applications in Higher Education [Online]. Hershey PA: IGI Global; 2013.
URL: <http://tinyurl.com/hh9b2qb>. Accessed 16 April 2014.

GHP client requirements

System

- OS (minimum)
 - iPhone 4
 - Android 2.3
 - Windows Phone 8, (WP 7, support only for important content)
 - Win 7
 - OS X & Linux, not specified
 - BlackBerry, not supported
 - Non phone devices (PlayStation, iPod, Kindle, ...), not supported
- Browsers
 - Chrome 15
 - Internet Explorer 9
 - Firefox 10
 - Safari 6.0
 - Latest Opera, support only for important content
- Mobile browsers
 - Native
 - Android 2.3
 - iOS 6
 - WP 8
 - WP 7, support only for important content
 - Other
 - Chrome, latest version
 - Firefox, latest version, support only for important content
 - Opera, latest version, support only for important content
 - Opera Mini, not supported

Devices

- Resolution
 - Full support for 980 x 550 and larger

- Important content and features need to work in 360 x 480
- Display size
 - Full support for 7 inch and larger (tablets and PC)
 - Important content and features need to work in 4 inch and larger (Phones and Phablets)
- Navigation mode
 - Cursor, supported
 - Touch and multi touch, supported
 - Focus, not supported

GHP courses will probably work in other systems too, but they are not tested nor function in them guaranteed.

[107]