Evaluation of JavaScript frameworks

Why should you use them?
Abstract
JavaScript is today’s most common client-side programming language for the web. Choosing the most adapted framework for each and every project can optimize the development process and increase profits. Developers often lack knowledge about what to prioritize when choosing a framework and very little previous research regarding this subject area is available. This thesis investigates two JavaScript client-side frameworks and answers the questions of why you should use a framework when developing applications. This thesis also investigates why a company should develop a new framework, when equivalent open-source alternatives are available. To be able to answer the questions interviews with developers were conducted. The results are that using a framework can optimize the development process regarding time and complexity, but learning a new framework can be difficult. In rare cases it is a great idea to develop a new framework for a certain project, but it is often not worth it. Developers also seem to choose experience of using a framework over performance.

Keywords: JavaScript, frameworks, client-side, Backbone.js, evaluation
Preface
I would like to thank my supervisor Jesper Andersson at the Linnaeus University as well as the supervisors at the company that I worked with for this thesis. It has been great working with you! Other important people have been my family, close friends and teachers that have shown their support and been brainstorming with me.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Background</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Previous research</td>
<td>1</td>
</tr>
<tr>
<td>1.3</td>
<td>Problem definition</td>
<td>2</td>
</tr>
<tr>
<td>1.4</td>
<td>Purpose and research question</td>
<td>2</td>
</tr>
<tr>
<td>1.5</td>
<td>Scope and Limitation</td>
<td>2</td>
</tr>
<tr>
<td>1.6</td>
<td>Target group</td>
<td>3</td>
</tr>
<tr>
<td>1.7</td>
<td>Outline</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Background and Theory</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>JavaScript</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Framework</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>Method</td>
<td>9</td>
</tr>
<tr>
<td>3.1</td>
<td>Scientific approach</td>
<td>9</td>
</tr>
<tr>
<td>3.2</td>
<td>Human centered approach</td>
<td>9</td>
</tr>
<tr>
<td>3.3</td>
<td>Analysis</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Results</td>
<td>12</td>
</tr>
<tr>
<td>4.1</td>
<td>General</td>
<td>12</td>
</tr>
<tr>
<td>4.2</td>
<td>Difficulty</td>
<td>12</td>
</tr>
<tr>
<td>4.3</td>
<td>Help</td>
<td>13</td>
</tr>
<tr>
<td>4.4</td>
<td>Time</td>
<td>13</td>
</tr>
<tr>
<td>4.5</td>
<td>Dependencies</td>
<td>13</td>
</tr>
<tr>
<td>4.6</td>
<td>Software</td>
<td>14</td>
</tr>
<tr>
<td>5</td>
<td>Analysis and Discussion</td>
<td>15</td>
</tr>
<tr>
<td>5.1</td>
<td>Problem solving/results</td>
<td>15</td>
</tr>
<tr>
<td>5.2</td>
<td>Method reflection and Reliability</td>
<td>17</td>
</tr>
<tr>
<td>6</td>
<td>Conclusion</td>
<td>18</td>
</tr>
<tr>
<td>6.1</td>
<td>Conclusions</td>
<td>18</td>
</tr>
<tr>
<td>6.2</td>
<td>Further Research</td>
<td>18</td>
</tr>
</tbody>
</table>

References | 19 |
1 Introduction

This chapter introduces and motivates this thesis. Previous research that has been made in this subject area are presented. The problem definition, scope and limitation and target group for this thesis are also covered in this chapter.

1.1 Background

JavaScript is today’s most common client-side programming language, used by 88.3% of all websites [1]. This requires that developers have knowledge about programming languages and the best approaches for implementing web applications when deciding upon which framework to use in a project. There is also many factors that influence the choice such as how easy the framework is to understand or how difficult it is to implement with. The development process can possibly be optimized and more profitable when using a framework, which makes this evaluation of JavaScript frameworks relevant not only for developers but project managers and other personnel as well [3].

I got the opportunity to work on this thesis together with a large international company. This company had a few years ago started developing a JS (JavaScript) framework that would adapt to their own requirements and their branding. Their main concern was the advantages or disadvantages of developing the framework with respect to the parts of the developing process that affected the developers personally. They also wondered how efficient and effective developing a new framework was compared to using an already existing open-source equivalent like Backbone.js.

1.2 Previous research

Previous research regarding JS frameworks has been carried out but tend to focus on performance and quality of web applications instead of the developers experiences of using the framework for developing applications. A study in 2002 measured the quality of web applications with the Web Quality Evaluation Method [2]. Plain JS, jQuery and Backbone.js was compared in a study conducted in 2015 but that study focused on performance and not the developers’ experience [4]. A study in 2013 compared the logic and performance of two JS frameworks (Angular.js and Backbone.js) but found no particular differences between them [5]. A comparative evaluation of JS frameworks was conducted in 2012, but the intention with that study was only to reveal drawbacks and provide help for the supporting communities of the frameworks [16].
1.3 Problem definition

The first problem to be investigated is how developers are going to be able to choose which framework to use when developing an application. The second problem is that some companies decide to develop a new framework for their intended applications instead of using an equivalent open-source alternative but they do not know if that is the best approach. The advantages and disadvantages of using a client-side JS framework will be discussed as well as the advantages and disadvantages of developing a new framework for a company, or an individual, instead of using an equivalent open-source alternative. The reason why some companies although decide to develop a new framework will also be discussed in this thesis. Companies and individual developers are interested in these problems and the answers, especially if the development process can be optimized when using a framework that is most suitable for each project.

1.4 Purpose and research question

The purpose of this thesis is to gain knowledge about client-side JS frameworks and how you as a developer choose the most suitable framework for each and every of your future projects. The reasons for and the utility of developing a new framework for a certain purpose will also be investigated.

The following research questions have therefore been identified:

- RQ1: What are the advantages and disadvantages of using a client-side JS framework when developing an application?
- RQ2: What are the advantages and disadvantages of developing a new framework for the development of an application, when equivalent open-source alternatives already are available?

1.5 Scope and Limitation

Not all available JS frameworks are evaluated in this thesis due to time constraints. The number of available JS frameworks increase every day and to evaluate every one of them would be a huge effort. The frameworks focused in this thesis are Backbone.js and the framework developed by the company. Only client-side MV* JS frameworks were considered during the selection process. The company was interested in a comparison of their own framework with an available open-source equivalent. This further decreased the number of available frameworks to choose from. Backbone.js was selected due to the fact that the company’s developers had some experience with that specific framework that would be an advantage during the comparison process. Backbone.js also fitted the requested prototype from the company really well, as it was a single-page application. Backbone.js was selected due to the fact that it is a popular and mature framework that has
been used by large companies and therefore feels reassuring for a company like the one I worked with. The selection of the framework to compare to the company framework was made in agreement with supervisors at the company.

1.6 Target group

The target group for this thesis are primarily developers working with JS that struggles with choosing the right framework for an application. Other developers, project managers and testers that want to optimize the development process in a project may also have an interest in this thesis. Students in the process of choosing a framework to master during their education may also have an interest in the results of this thesis, as well as the teachers that will educate students in the development of applications for the web.

1.7 Outline

Chapter 1 introduces this thesis and provides the reader with the required background information for this study. This is also where the problems are defined and presented, as well as the research questions. Chapter 2 gives the reader knowledge required for understanding this thesis and the problems that may occur. Chapter 3 describes the method of this thesis, including the scientific and human centered approach. The selections for the interviews as well as the design of the interview questions is covered in this chapter. Chapter 4 presents the results from the study conducted. Chapter 5 analyses and discusses the received results and answers the research questions. Chapter 6 presents the conclusion of this thesis and some thoughts about future research.
2 Background and Theory

This chapter covers the required theory for the understanding of this study. JavaScript and the selected JS frameworks will be explained at a high-level approach. Statistics showing the popularity of JS frameworks will also be presented and the most popular frameworks will briefly be compared to each other.

2.1 JavaScript

JavaScript is today’s most popular programming language for the client-side programming of all web-applications [1]. JavaScript was developed already in 1995 by Brendan Eich. The syntax of JS was taken from the C language but was simplified by Erich and was intended to be a more friendly language than the available alternatives. The first version of the JS language was developed by Eich in only ten days but has since that version been updated. The first version of JS has inspired several other projects that later on have extended the many possibilities of JS [6]. The first version of JavaScript was released in 1996 [7].

2.2 Framework

A software framework is a set of source code and libraries that provides help to the developer when coding. A framework can help with a lot of different aspects of an application such as data storage, security and user interfaces [8]. For example a security framework can be used for an application. The developer will then not need to implement the authentication and authorization functionality since it is already implemented and provided by the framework. The developer will only need to use the provided component that has the required functionality, and maybe make smaller changes to make it fit the application. To simplify it even more a person does not need to invent the wheel to be able to build a soapbox car, she just have to install it.

A web application framework is a type of a software framework that helps developers specifically when developing web applications [9]. There are many different frameworks available that all are designed to fulfill some requirements. Frameworks can be applied to many areas such as education or healthcare, but in this thesis the focus is on computer science and more specifically web programming.

The frameworks can be divided into different categories that uses different patterns. One set of patterns are MV* (Model View *), which separates the model, the view and the logic in an application. Another pattern is the Template Pattern in which an abstract class define the way to execute methods or the Factory Pattern were we create objects without exposing the creation logic [18][19].
The popularity of JS frameworks is important for the developer that is going to choose a framework to master. A popular framework will most certainly have more tutorials and help will more easily be found online. The most used JS frameworks are presented in Figure 2.1.

![Top 10 Most Used JavaScript Frameworks](https://blog.codeanywhere.com/top-10-most-used-javascript-frameworks/)

**Figure 2.1: JavaScript frameworks usage statistics.**

AngularJS, Backbone.js and Ember.js are the top three most used JS frameworks [14]. They are all similar to each other and use the concept of views, events, data models and routing. Ember.js is the oldest framework, developed in 2007. AngularJS and Backbone.js is a few years younger, developed in 2009 and 2010 respectively. All these frameworks are represented in online communities where developers can provide help to each other and find solutions. They can also handle templates, but AngularJS does not need any external engine to do this. AngularJS is also the only framework that does not need any external library to work properly [15]. This results in that the size of AngularJS does not increase when the required dependencies are taken in consideration. The size of the frameworks are mostly what separates them.
The size of the three most used JS frameworks are presented in Table 2.2:

<table>
<thead>
<tr>
<th>Framework (version)</th>
<th>Net Size</th>
<th>Size with required dependencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>AngularJS (1.2.22)</td>
<td>39.5kb</td>
<td>39.5kb</td>
</tr>
<tr>
<td>Backbone.js (1.1.2)</td>
<td>6.5kb</td>
<td>43.5kb (jQuery + Underscore)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20.6kb (Zepto + Underscore)</td>
</tr>
<tr>
<td>Ember.js (1.6.1)</td>
<td>90kb</td>
<td>136.2kb (jQuery + Handlebars)</td>
</tr>
</tbody>
</table>

Table 2.2: Framework sizes

Table 2.2 shows that even though Backbone.js is the smallest framework the size of it grows a lot with the required dependencies, something that AngularJS does not. The size of Ember.js increases the most. jQuery, Underscore, Zepto and Handlebars are all external libraries used by JS frameworks.

To summarize the comparison of these frameworks Ember.js is the oldest and also largest framework, Backbone.js embraces minimalism and AngularJS attracts pure web programmers since it extends the HTML dialect but the differences between them are hard to find even though they matter [15]. Because it is so hard to find the differences between the frameworks every intended project has to be evaluated carefully before a framework is chosen.

2.2.1 Backbone.js

Backbone.js is an open-source JS framework. It is a light-weight framework, which means that it has a low impact on the actual application. Backbone.js only has one dependency to an external JS library (Underscore.js) which also makes it easy to install and use [10]. Backbone.js was created by Jeremy Ashkenas in 2010 [11].

Backbone.js has a website where interested developers can find information, tutorials and help. Even though there is no online community specifically for Backbone.js there are active online communities with developers that help each other without caring about the programming language or which framework that has been used. In these online communities it is the developers, as users, that both ask the questions and answer them. Stackoverflow.com is considered the largest online community and offers a lot of help to developers all around the world [13].
Backbone.js is divided into four different parts: models, collections, routers and views. These are briefly described in the following sections. The framework is especially suitable for developing single-page applications [3].

2.2.1.1 Models
A model is the part of the application that contains the actual data. The model also contains logic such as validations and access control. The model is the actual heart of an application, and without it the application will not work [12]. This is a required part of all MV* frameworks.

2.2.1.2 Collections
Collections are the ordered lists of models that help with all interactions with the models that are used in an application. Events can be bound to the models, and listeners then listen to these certain events and when an event occurs a specified action is performed [12]. For example a model’s name can be updated when a change in a form is made. The changed data will be saved in the model and be available to use later on in the application.

2.2.1.3 Routers
Routers are the part of the application that provide routing methods. They also provide methods for connecting the client-side pages with actions and events. Routers make it possible to use standard URLs instead of hash fragments for linkable and sharable URLs in the application [12]. This is very helpful when developing a multi-page application where you want to access or bookmark a specific page.

2.2.1.4 Views
Views are used to organize the interface of an application into logical views that can be updated without having to redraw the entire page. They although do not determine any HTML or CSS. The render function of a view can be bounded to an event that updates the HTML when a change occurs [12]. For example a header can be edited without the need to redraw the entire article. Views are parts of every MV* frameworks.

2.2.2 Company framework
The company framework is protected by confidentiality. The developing process of the framework started in April 2013 by a team of developers,
employed by the company. A full team is today working full time on the framework with development and maintenance. The framework is a full architecture stack framework that includes all required parts of an application, such as networking, user interfaces and business logic. It is based on small, independent and reusable components, designed to be easy to scale, maintain and upgrade. The framework also includes a separate framework for the styling of an application, developed specifically to fulfill the company's brand and provide a consistency between their applications. Most parts of the framework are very similar to Backbone.js, such as the fact that the frameworks are designed for single-page applications. The frameworks can be found to be equivalent to each other, except the included styling framework that the company provides.

The core framework is divided into different components. The main controller has the main responsibility for the other components in the application. The framework then includes interfaces that the main controller uses to share common context between completely independent instances of the application. These independent instances manage specific well defined responsibilities and can be removed or restarted without affecting any other instance. The framework also includes components that consist of reusable generic user interface code. These components do not have any knowledge of their own context or how they are being used. These small components can also be reused in multiple applications and can be combined to provide a common user interface component. The framework also includes a view that is responsible for the visual components of an application, which is a part of all MV* frameworks.
3 Method

This chapter describes the method used in this thesis. The method is divided into two parts: a human centered approach with interviews and a scientific approach with implementation of software. This chapter covers the selection of interview respondents and the design of the interview questions. An analysis of the used techniques finishes this chapter.

3.1 Scientific approach

For this thesis an inductive approach has been chosen. A tentative hypothesis was given by observation of developers and provided the foundation of this study. This tentative hypothesis has then been worked on to be able to be proved true or false. Qualitative data was given from the study conducted since the number of available respondents for the interview were not enough to produce quantitative data. The interest was to get personal experience from a smaller group of developers.

3.1.1 Software development

To gain knowledge and hands on experience about Backbone.js a given prototype from the company was implemented. The developer had no earlier experience of implementing Backbone.js. The prototype was implemented together with the company and one of their developers after a request from the company.

The prototype developed is a CAi3g provisioning client that are designed to work with an existing product from the company. CAi3g is a flexible protocol that follows the standards used for web services technology such as SOAP and XML. It's aiming for providing a simple, up-to-date and unified provisioning interface [17]. The purpose for this prototype was to replace a complex software that today are used by the company. The software is used for doing a specific task but in a very difficult and confusing way. The task to be completed is to add and change attributes to a certain product or service. The prototype is protected by confidentiality, but it was the experience from developing the prototype and not the actual code that was relevant for this study.

3.2 Human centered approach

An interview with developers at the company was conducted. The interview was conducted with a small group of developers to encourage discussion and make it possible for all respondents to get their opinion heard. The discussion opportunity would have been lost if the interviews were performed
individual. If the interview had been performed with a greater number of persons some opinions may have been lost.
Another available technique for gaining the developers’ experiences would have been to perform a survey but the thought of a survey was rejected early in this thesis. The interview was supposed to cover personal experiences of each one of the developers. This purpose may have been lost if the experience had been written down instead of spoken. The interviewer now got a chance to get a deeper understanding of the answers. The interview was also intended to encourage discussion. A survey would not have achieved the goals for the intended interview. If the number of available respondents had been largely increased a survey should have been reconsidered due to the extra work load.

3.2.1 Selection
The selected developers had varying experience of developing with JS frameworks. The criteria when selecting respondents was that the respondent should have experience of working with plain JS as well some experience with as at least the framework developed by the company. The respondents also had to work on the company with the company framework on a daily basis. This decreased the available respondents severely, and although all developers that fulfilled the criteria participated in the interview the respondents were only three.

The selected respondents were between 23-24 years old. They had all studied three years at the same university, in very similar study programs, and had worked at the company for a few years. One of the respondents did not have any JS education from the university and the other two had only studied JS for five weeks each at the university.

The respondents had between half a year and two years of experience of JS development. One of the respondents did not have any other experience of JS frameworks except the company framework. The other two had worked with JS frameworks also outside of work. The respondents worked in different teams of the company and on different projects but had worked together at some point during their employment.

3.2.2 Design
The interview questions were constructed based on experience from developers, including me, and relevant literature. From the experiences and the literature common problems and situations were picked. These turned into concrete interview questions, supposed to include the most important parts of the developing process. The interview questions covered subjects from the choice of a development editor, to the actual implementation process and the programming language itself. Most questions were open-ended to encourage
discussions, but some questions were also close-ended and answered by a number that represented how the respondent felt about a certain statement. The interview questions also included general information about the respondents such as gender, age and education.

3.3 Analysis
The interview were conducted and the answers were written down during the interview. Summaries of the answers from the interview were then made, focusing on every respondent individually. The summaries were then compared to each other in order to find experiences that were similar, or really differed. This made it possible to find patterns in the answers. The answers provided knowledge about the developers’ experiences with implementing a JS framework. The experiences then worked as the base for the evaluation of the frameworks. The given experiences were compared with the gained experiences from developing the prototype and the similarities and differences between the experiences were then discussed. The discussion covered the situations from the interview and provided the basis for answering the given research questions.
4 Results

This chapter presents the results given from the interviews and the developed software. The chapter is divided into sub-chapters following the interview sections.

4.1 General

The interview showed that the developers felt like they did not need a framework for being able to implement a software. The framework provided a lot of help but was considered more to be a helpful tool, not a necessity. The most time consuming part of implementing an application for a company was found to be the styling part. The styling often had to fit in with a specific branding and be consistent. If the branding was provided as framework and language independent no other framework was considered necessary.

The office were the interview was conducted has several hundred of employees but the age, gender and education of the respondents were found to be very similar. Also how many years the respondents had worked at the company did not really differ.

4.2 Difficulty

The interview showed that the company framework is a quite simple framework to use. The problems appeared when trying to implement something that the framework did not support and the framework did not have a great support even though it had increased in the last couple of months. The framework was perceived to be harder to work with than it was difficult to understand. Backbone.js provided less help than the company framework, but both frameworks provided more help than plain JS.

The interviews showed that the produced code for the company framework was easy to understand, but that the easiness of understanding the code also depended on the developer that had written it. The company framework also provided code that was easy to understand with clear method names and a good structure. The more complex the application was, the more difficult it was to understand the code. This was found to be applicable to both frameworks.

How easy it was to understand and implement a framework was found to be very important for developers when choosing which framework to use. The interview showed that this was more important for the developers than the actual performance of the framework. The frameworks today was perceived to be performing at the same level and the small differences was found to be irrelevant.
4.3 Help

The company framework was found to not have any online community where developers could provide solutions for each other or ask for help. It although had an online forum where questions could be asked. These questions were answered by the developers that are developing the framework and it could take a considerable amount of time before the questions were answered. There was no available online information about the framework that the developers could search for, only the given tutorials from the company. Backbone.js was found to have a lot of online information as well as being part of online communities where developers could help each other.

Both frameworks had several editors that helped the developers with suggesting code and auto completion when implementing. This was not found to be any problem or something the developers considered when choosing a framework. The developers mostly had a favorite editor that they always used, regardless the framework, if they could decide for themselves. At work they sometimes had to use a specific editor, and the developers always had to use the company framework when developing in JS.

4.4 Time

The company framework was found to have many ready components that the developers could use instead of implementing them again. These components provided functionality and was perceived to be more than in Backbone.js. This decreased the required time to implement a feature compared to implementing it in plain JS. The time spent on implementing a feature was also the same when implementing in the company framework and in Backbone.js. The company framework was found to have a high learning curve since there was no online community offering help or solutions. Although Backbone.js was perceived to be easier to learn it was also found to be harder and require more time when implementing after you have learned the framework.

The required time to learn and master a framework was found to affect the choice of a framework. Also the required time to implement a framework was found to be considerably important.

4.5 Dependencies

The company framework was found to have very few dependencies. The framework itself provided help for the installation process of the dependencies and the developers did not have to configure anything by themselves. The dependencies required was found to be less than for Backbone.js and also easier to install.
The complexity of the dependencies was not found to be prioritized when choosing which framework to use. The problems with dependencies was a later issue that caused a lot of problems but still that was not considered when choosing a framework.

4.6 Software

The development process of the given prototype resulted in a simple prototype that fulfilled the requirements from the company. The development of the prototype also gave many experiences. The given experiences from the development process were that it takes a lot of time to learn a new framework, and even more time before you can master it. The number of stakeholders in a project was also found to make the development process more complex.

The confidentiality concerns was found to make the development process harder since it was not allowed to get help from other developers with some problems that revealed information about the company.
5 Analysis and Discussion

This chapter analyses the given results in chapter 4 and discusses them as well as the implementation of this study. The research questions are also answered in this chapter.

5.1 Problem solving/results

The given results lived up to most of the expectations, but the expectations were based on others’ personal thoughts since no relevant scientific evidence from previous research could be found. This means that the common thoughts you can find online and by asking other developers seem to match the results from this study.

The results showed that developers prioritize the development experience over the actual performance of the application. How easy a framework is to read and understand is really important but how fast the application are is unimportant. It is also considered to be very important that the framework is easy to implement with and that help is available online from other developers.

RQ1: What are the advantages and disadvantages of using a client-side JS framework?

The advantages of using a framework was found to be that it takes less time to implement an application when using a framework, due to the fact that the framework provides help that plain JS does not. The developers does not need to repeat themselves or implement functionality that is already provided by the framework. This was an expected result that all developers agreed on, and it matched the expectations I personally had.

The disadvantages of using a framework was found to be that it takes time to learn and master a framework before you can use it properly. If the developer is not familiar with a framework plain JS may therefore be more efficient. This depends on how advanced the functionality of the intended application is. A high learning curve means that the developer needs to spend a lot of time learning the framework before it is usable. If the intended functionality is very basic the framework may not decrease the time spent on the development process if the learning curve is very high. The impact the learning curve had on the choice of a framework was somewhat surprising and not considered before this interview was conducted.
RQ2: What are the advantages and disadvantages of developing a new framework for the development of an application, when equivalent open-source alternatives already are available?

The advantages of developing a new framework for a company are that the company can design it to fulfill all the given requirements and also to include a specific branding. This saves a lot of time for the developers when implementing an application. A new framework can also define which external libraries that can be used which can improve the safety of an application. This could be a major concern at large companies with confidentiality concerns, but maybe not considered important at all by smaller companies. This would not be an advantage for an individual developer.

The disadvantage of developing a new framework are that it can take a lot of time to develop a new framework and it is not always worth it. If the applications that are intended to be implemented are not advanced or does not need a specific branding the time spent on developing a framework will most certainly be a waste. The need for developing a new framework will have to be evaluated carefully and the choice cannot be made easily. There are many factor that come into play during this kind of choice, such as security, budget and the developers’ skills.

In this thesis the differences in the experience of using Backbone.js versus the company framework were found to be very small. The intended functionality could be implemented in both frameworks. The only thing that differed was the styling framework that Backbone.js obviously did not have. Both frameworks were easy to read and understand, and the possibility to continue working on written code by other developers depended on the actual developers and not the framework. The advantage of developing a new framework could be that the company gets more control over the external libraries that are used and a consistency could be achieved. This could although also limit the developers in their development process as well as the possibilities of developing in new creative ways. The dependencies could also be limited and ease the startup process of a project, but since this was not considered important by the developers in the interview this would not count as an advantage from the developers’ point of view.

No major differences between the frameworks could be found and no major advantage of developing a new framework for the functionality of an application could be found. The advantages was found to mostly affect the styling part of an application, and this could lead to that companies develop frameworks that fulfill their branding but that are compatible with other open-source frameworks that handle the functionality.
5.2 Method reflection and Reliability

The interview would have been more reliable if the number of respondents had been higher. The lack of available participants did not make the result of the interview less reliable, since all developers that fulfilled the criteria participated a more reliable result was not able to be retrieved in this study at this time. For a greater reliability this study would have to be performed repeatedly with new developers, which had gained the knowledge and experiences required until a greater number of answers could have been retrieved. If other similar companies also had developed and worked with a similar framework those companies could have been included in this thesis. That was although not investigated.

The answers from the interview can also have been misunderstood, even if that risk are lower during an interview than in a written survey. The questions can also have been unclear, but the respondents was given the opportunity to ask the interviewer if they found anything unclear. If a survey had been conducted instead of the interview the questions most certainly would have been harder to understand and more misunderstanding would have occurred.

The answers might have been shaped by the respondents to make the company framework seem better than it actually was, or by the interviewer to match the preconceptions. To reduce the risk of the respondents shaping their answers the respondents was not made been aware of what study the interview was a part of. The risk of the interviewer shaping the questions and answers to match the preconceptions is hard to avoid.

If the prototype would have been implemented in both Backbone.js and the company framework the prototypes could have been compared to each other and results from that comparison could have been presented. That could have given a better understanding of the differences between the frameworks, but that was not possible to do due the extra amount of time spent on the confidentiality problems with the company. I did not receive the documentation that would have been required for the implementation in the company framework and the supervisors at the company did not have to time to implement the prototype together with me. The purpose of the prototype was although to gain experience from implementing Backbone.js and the results should not have been affected. The confidentiality concerns may also have affected the results of this study due to the fact that not all documentation and information was available for me at the time when I needed it. This may also have affected the respondents’ answers.
6 Conclusion
This chapter presents the conclusion that have been made from this thesis and some thoughts about further research.

6.1 Conclusions
The results showed as expected that there are both advantages and disadvantages of using a client-side JS framework. The advantages were more than the disadvantages and the usage of a framework can therefore be recommended. The major advantage of using a framework was the provided help and the decreased time required for the development of an application. The major disadvantage was that the framework may be hard to learn and master. If the intended application is very basic the effort it takes to learn a framework may be greater than the effort to implement the application in plain JS. The advantages for a company to develop their own framework was found to depend on the complexity and size of both the company and the intended application. If the company was small and no confidentiality or security concerns existed the disadvantages of developing a framework were more than the advantages, especially if the intended application was very basic. The conclusion that have been made from thesis is that frameworks mostly optimizes the development process, but that the need of developing a new framework for a company may not exist.

6.2 Further Research
Further research that includes interviews with a larger number of respondents and with evaluation of more client-side JS frameworks should be performed to get a deeper understanding of the impact the choice of frameworks can have on the development process. This thesis is just a first attempt to discover what developers consider important when choosing a framework to use and should therefore be extended in the future. Also further research that includes other types of frameworks could be performed. If the experience of working with frameworks differ due to which programming language that is used could also be interesting and relevant to investigate. If some programming languages are more dependent on using a framework, and using the best fitted would be a great knowledge for students and teachers when considering using a framework in a course.
References


