Comparison of Browser Load-Time Performance and Ease-of-Use for JavaScript Web Development Environments

Zachary Purdy

I. Abstract

In the rapidly growing field of web development, a plethora of development options exist for new and aspiring programmers. Because of so many options, new developers can be paralyzed by what they should be learning. Angular and Ember, two JavaScript based frameworks, are tested against React, a JavaScript library, to see differences in load-time performance in a few different browsers to help people in this position choose one that best fits their needs. A simple, "Hello, World!" program is written in HTML to ensure minimal coder influence and to keep focus on the performance of the actual platforms. While all three platforms don't perform radically different than each other, it's made clear that there are differences in performance that could be magnified on a larger project scale. However, it's important for newer developers to remember that there is never a one best solution to a problem. This research should help to inform someone of possible choices and make certain limitations aware, not conclude a clear "winner".

II. Introduction

As technology constantly progresses, it's easy to assume that the field of computer science will always be changing to accommodate the needs of developers, users, and customers. The web development industry is a flourishing sub-field that is always breeding new solutions to old problems. Each year, new web development platforms are created, and old ones are modified, to satisfy the needs of the developers in this industry. One of the most common programming languages, JavaScript, is used alongside HTML and CSS to create the perfect trifecta to develop web sites and web applications. It's accessible and mostly easy to learn, causing many young developers to be fighting for a piece of the pie that big businesses once primarily held. With the rise of the Internet of things and integrating countless network connections in our lives, there will certainly be plenty of opportunities for years to come. According to the U.S. Bureau of labor statistics, the web development industry is expected to grow at a rate of 27% within the next decade [1]. However, with ever evolving demands and technology comes more ways to learn to develop for this industry.

While pure JavaScript (vanilla) is a fine language for writing scripts, different communities and companies have developed many frameworks over the years to meet their individual needs. Frameworks are specialty environments built from JavaScript that become the new development environment. JavaScript is still used, but usually with some tweaks. For example, Angular uses a Model-View-Whatever styled approach where there is a Model, a Viewer, and whatever works for you afterwards such as a Controller or a View-Model [9].

Angular also incorporates two-way data binding and dependency injection [2]. Vue.js, on the other hand, is great for fast cross-platform needs. There is never a one "best" option for every job though. It can be confusing to young/new developers who aren't sure what to learn, yet are struggling to break into the industry whether by freelance or by applying to tens or hundreds of different jobs [6]. It appears that every company has an idea of what they feel is the right environment to use, making it tough on these young developers to keep up and constantly learn something new. Even JavaScript is not always desired as other options exist for web development such as Python, Java, PHP, and Ruby [4], [5]. This can cause paralysis by analysis; spending too much time debating what to learn, rather than just learning something and getting experience under the belt.

In this paper, the research will present a comparison of three popular JavaScript development environments. Similar research has been done in the past where frameworks were tested on maintainability, complexity, and size metrics. Results showed that the researchers wanted to find the best framework, but there were many factors involved with no clear "winner". Each demonstrated several pros and cons for the tests performed on them. The material has since become outdated and focused on the frameworks themselves at the time, rather than projects built with them [3]. This will assist in helping a developer choose a development environment that more closely fits their needs, yet at the same time showing that there is no best tool for a job. Strengths and weaknesses of these frameworks will be shown through load-time performance benchmarks, ease of use, and resource consumption. The three environments to be used will be Angular 2, a framework promoted by Google, Ember, an open-source framework used by companies such as Twitch and Netflix for their online applications, and React which is a library originally developed by Facebook.

Which framework used more lines of code? Different frameworks have different ways of implementing code even though they are still JavaScript under the hood. Which one is taking up more space to implement? Line count, folder structure, and overall directory size are important metrics to consider. Which one loads faster in different web browsers? If a website is slower than a competing website by several seconds, users may abandon it for the competitor. Benchmarks show which environment delivers a faster load time under certain conditions.

New developers will want as fast and responsive of a system as possible when learning and testing new code. Programming can be frustrating as it is, and an impatient web developer will simply get frustrated if their code consistently takes too long to load. This data will transfer well into the mindset of the average Internet user as well. No one wants to wait for web pages to load, as our society is very centered around instant gratification. Sometimes, the difference of a few seconds can mean one business losing customers, while their competitor's gain new ones.

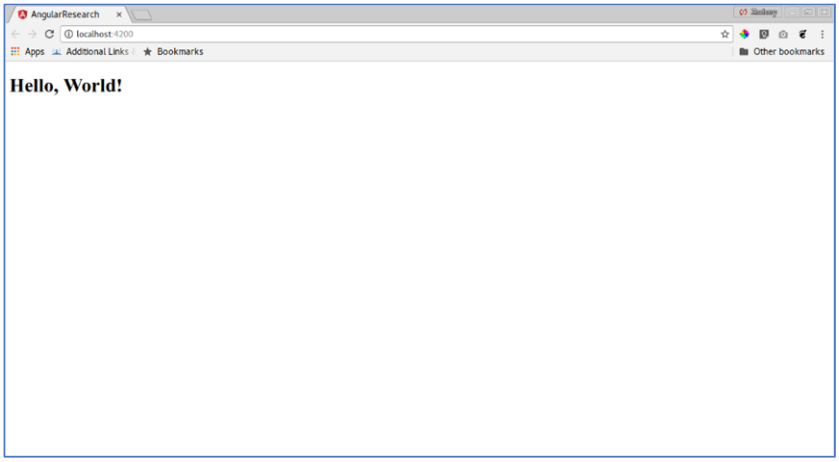
III. Materials and Methods

To show the comparisons between the different JavaScript environments, similar websites have been developed in each. A simple "Hello, World!" application is run with the phrase printed across the top of the page using HTML. Google Chrome 63.0.3239 and Firefox Quantum 1.0 are

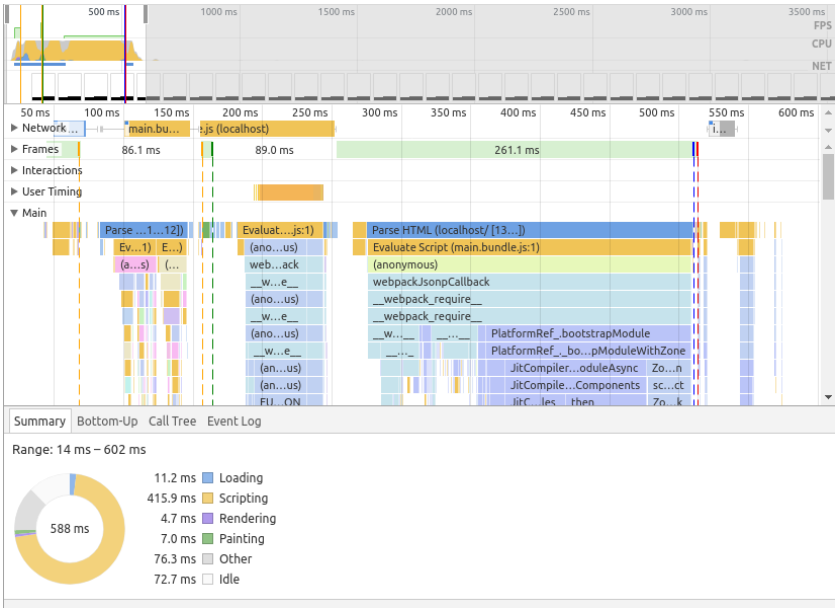
used since they have developer tools that can actively measure load times, performance, memory, security, and more. These tools have a lot of in depth features, but for now, performance / load times will be focused on. Performance benchmarks have been taken under the same conditions, running on an Ubuntu Linux Laptop manufactured around 2013 with no programs running, except a terminal instance to launch the web server and the browser currently being examined. Visual Studio Code 1.18 is being used as the code editor. A GitHub repository is used as a repository. The aspect of the websites that have been tested are the load times of the pages on the various browsers.

IV. Results

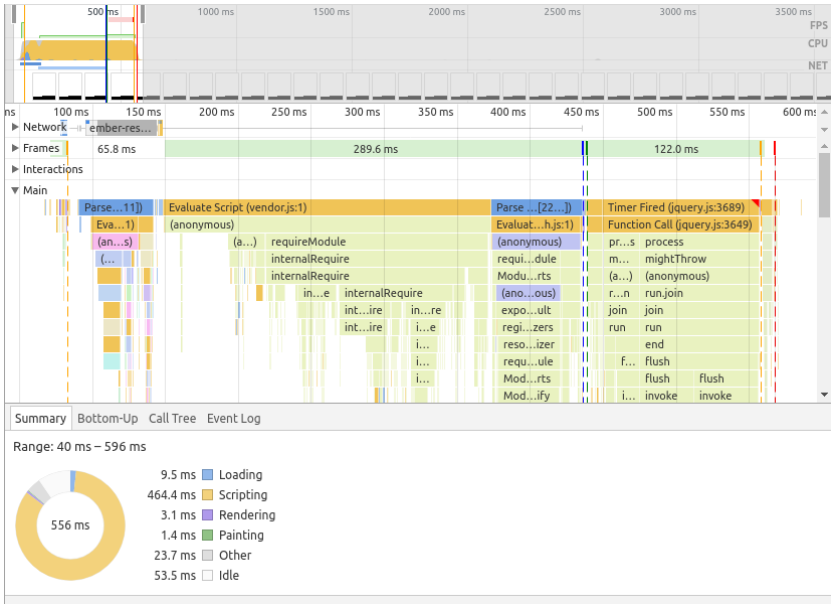
It can be observed that there are some clear differences in load-time performance between a site developed in Angular, with similar sites built in Ember and React. Figure 1 shows what this web page looks like in the Angular environment. The Ember one is identical and the React application has text in the middle, but it is still displays the same text.

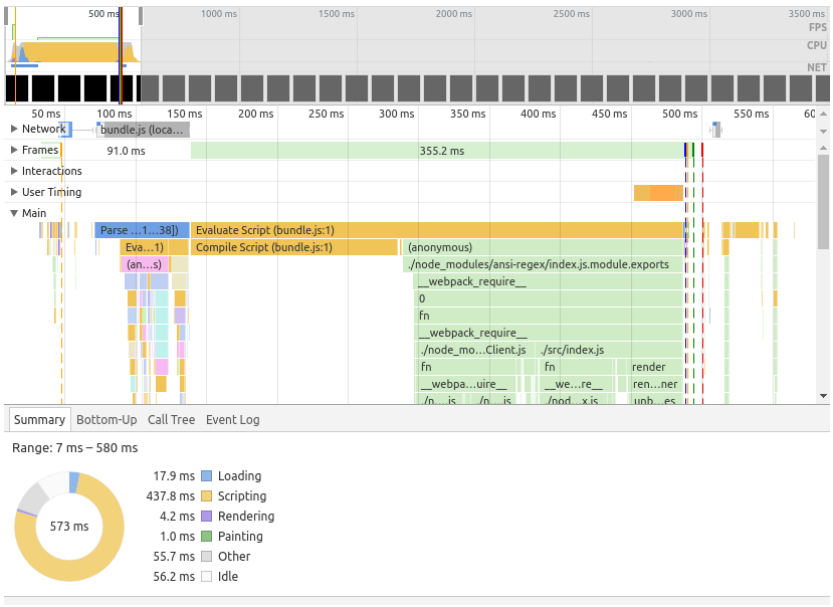


[Figure 1: An example of what is being loaded into the browsers]



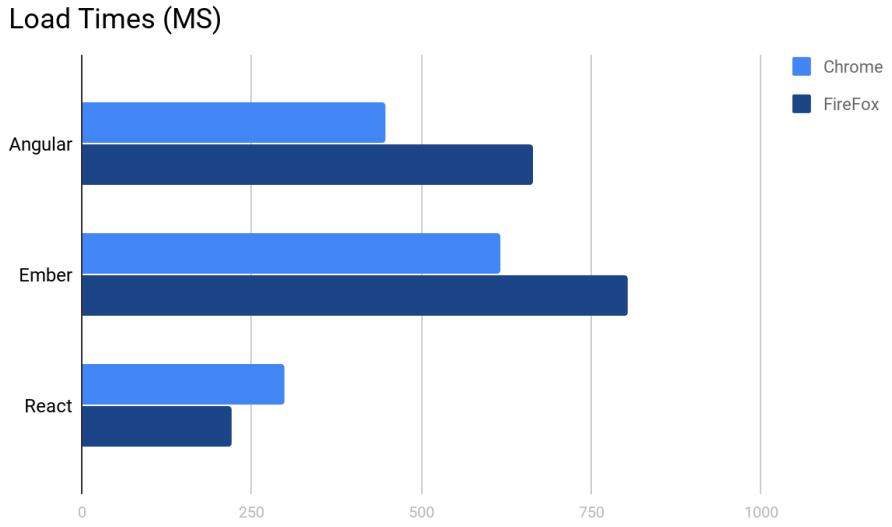
[Figures 2,3,4: In order, Angular, Ember, and React sites in Chrome Dev tools]





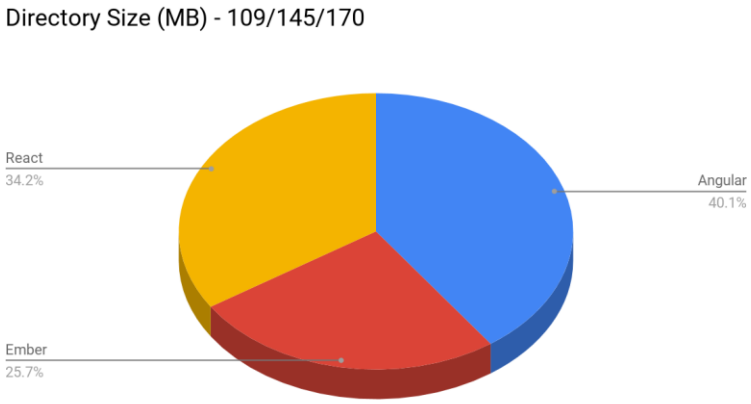
It can be observed from the Chrome Developer Tools screenshots that each environment is made up of very different components. No two frameworks/libraries will run the same as they have been developed to meet certain needs by the groups that made them. The scripting portion of the sites is always the main consumer of time in these load tests. The same scenarios hold true for the Firefox Developer Tools.

The following load-time results can be observed in Figure 5. The averages are based on thirty ‘runs’ per webpage per browser.



[Figure 5]

Another important area to pay attention to is consumed resources. Hard drive space may be cheap, but when considering an ever-evolving project, memory space consumed would certainly have an impact down the line. The local machine is not the only concern as saving time is always a valuable thing with uploading new builds to GitHub or deploying a new build of the site. Figure 6 shows the resources consumed for this machine.



[Figure 6]

V. Discussion

It can be expected that a vanilla JavaScript will handle tasks simpler and faster. There is no clutter to the code. An HTML index file references the script and it runs. Frameworks are created to solve specific problems. However, they get in the way of development, especially with newer developers. For the scope of this project, it is not necessary to discuss the more advanced features of Angular, Ember, and React. Newer developers should be focused on learning core programming principles and how they relate to JavaScript. At this stage, frameworks and libraries are only an added layer of complexity. Performance of the local system and the site also take a hit as an environment is being used to display data that doesn't even take advantage of the advanced features of a chosen framework.

Between the three choices presented in the article, it can be observed that React came out as a clear winner in both browsers. This could possibly be due to the fact it is a library and not a framework, therefore the classification might mean it was developed entirely differently. Ember took the longest out of the three, coming close to the one second mark in Firefox. Angular performed second best overall, which may be a testament to its popularity. Not everything can be tested however. In this research, only some simple HTML was coded onto the page. While this is a starting point, future work would have to be done properly implementing DOM manipulation features or scripting to show how code is effected across the three platforms. In the research of resource consumption, Ember is shown to be a more lightweight solution than Angular and

React. It's important to note that the actual webpage that was loaded could be done in only a couple of lines in a plain text file. These webpage directories contain the framework and library code as well. This hard drive space consumption could certainly be reduced by not using the full source files of the environments, and rather have only referenced the code in the header files. However, this can present the problem of the sources disappearing from the web, causing the page to not be able to load.

Other papers have researched comparisons of web frameworks in the past [3], [5]. The first one is outdated, but especially important. There is software that can analyze the core code of the frameworks to determine efficiency, maintainability, and errors. Unfortunately, most of this software is outdated or no longer maintained. Useful metrics were tracked such as maintainability, size, and complexity. The testing was done on the actual framework code, not on applications built with them. For future research, it would be useful to find software that would analyze modern frameworks in the same way. Community involvement is important, especially in an open-source tool like Ember, and analyzing these tools leads to healthy competition and improved experiences for all users and developers.

Another way this research could be approached would be to use a case study [7]. However, these cannot always be a reliable source of research as they can end up coming from too narrow of a perspective. Surveys can also be a useful tool to judge the popularity and effectiveness of frameworks, as seasoned web developers have plenty of experience. However, surveys can present sample bias problems and only give opinions as results, rather than data that is more useful and concrete [8]. While these other methods are not ideal, they may be utilized to aid in the research process. It's important not to completely dismiss a possible tool.

Two major limitations exist for this research project. One is that the research is based on one person's coding ability, rather than what the actual efficiency could be for the frameworks. Perhaps with more time, better code would be produced that could change the results of load times on certain browsers. In some preliminary tests, the Ember website loading much faster than the Angular site with over 50% improvement. When the code was implemented in another way that followed the way Ember was supposed to be structured, performance suffered greatly as the application had to go through a longer process to get to the code. In the original planning stages of the project, the intention was to test performance of some JavaScript code in each environment. However, knowledge of each one and coding ability would've skewed the performances. Therefore, the testing was switched over to using a simple, "Hello, World!" statement as this would not influence performance. Instead, load-time testing was done only with the barebones websites. Because of this, Angular, Ember, and React were all tested with less coder influence. This provides a more accurate assessment of the individual environments.

The other limitation is time to learn different programming languages. For the purposes of this research, two frameworks were compared to each other. In a perfect world, there would be time to make the same website in other environments such as Python with Flask and Ruby on Rails to compare a totally different environment. Some other useful metrics to observe would be loading the same site through a popular online building service such as WordPress or Wix.

VI. Bibliography

[1] "Web Developers: Occupational Outlook Handbook: U.S. Bureau of Labor Statistics." [Online]. Available: https://www.bls.gov/ooh/computer-and-information-technology/web-developers.htm. [Accessed: 29-Sep-2017].

[2] M. Ramos, M. T. Valente, R. Terra, and G. Santos, "AngularJS in the Wild: A Survey with 460 Developers," in *Proceedings of the 7th International Workshop on Evaluation and Usability of Programming Languages and Tools*, New York, NY, USA, 2016, pp. 9–16.

[3] A. Gizas, S. Christodoulou, and T. Papatheodorou, "Comparative Evaluation of Javascript Frameworks," in *Proceedings of the 21st International Conference on World Wide Web*, New York, NY, USA, 2012, pp. 513–514.

[4] P. Neves, N. Paiva, and J. Durães, "A Comparison Between JAVA and PHP," in *Proceedings of the International C* Conference on Computer Science and Software Engineering*, New York, NY, USA, 2013, pp. 130–131.

[5] J. Ignacio Fernández-Villamor, L. Díaz-Casillas, and C. Á. Iglesias, "A Comparison Model for Agile Web Frameworks," in *Proceedings of the 2008 Euro American Conference on Telematics and Information Systems*, New York, NY, USA, 2008, p. 14:1–14:8.

[6] L. Mannila and M. de Raadt, "An Objective Comparison of Languages for Teaching Introductory Programming," in *Proceedings of the 6th Baltic Sea Conference on Computing Education Research: Koli Calling 2006*, New York, NY, USA, 2006, pp. 32–37.

[7] D. E. Perry, S. E. Sim, and S. Easterbrook, "Case Studies for Software Engineers," in *Proceedings of the 28th International Conference on Software Engineering*, New York, NY, USA, 2006, pp. 1045–1046.

[8] L. A. Meyerovich and A. Rabkin, "How Not to Survey Developers and Repositories: Experiences Analyzing Language Adoption," in *Proceedings of the ACM 4th Annual Workshop on Evaluation and Usability of Programming Languages and Tools*, New York, NY, USA, 2012, pp. 7–16.

[9] "MVC vs MVVM vs MVP. What a Controversial Topic That Many Developers Can Spend..." Accessed December 10, 2017. https://plus.google.com/+AngularJS/posts/aZNVhj355G2.