First time contributors guide for front-end frameworks

Team 7

Akshay Prasanna Kunal Satija Jose Cruz

Stevens Institute of Technology

December 10, 2021

1 Abstract

As a new developer, you may be wondering how to contribute to an open-source project, whether because you have a particular interest in open source or because it's a great way to build a portfolio for potential employers.

There has been an increase in demand for front-end developers in the last year, so here are some pointers to get you started. It also includes several frameworks from which to choose.

We hope that by publishing this paper, we can provide some insight into the most popular frameworks (react, vue, angular, and svelte) and their communities.

After all, one of the fundamental pillars on which open source projects can thrive is a strong community.

2 Introduction

There are multiple open source projects out in the open, specially on Github, which house over 100 million projects [1], but we want to narrow our scope to the front-end because is a easy path to introduce a new developer and also the impact of the collaboration will affect the lives of multiple developer and even more, the users of this applications.

In this paper we are going to focus on the main four front-end frameworks, this paper it target towards potential first time contributors, and which to provide a background on how the communities of this projects interact with each other and the project.

3 Background and Motivation

3.1 Background

When a developer wish to starts it's journey as a contributor, they don't know how to begin, we suggest starting with one of the front-end frameworks because it not only increase the developers skills, but with the interaction with other people it also provide access to this developers to interact with developers with far more experience which in turn will help them to find more opportunities outside of the project.
All the data was collected from github up to November 19, 2021

3.2 Current state of the world

We use the top four most used front-end framework: **React**, **Svelte**, **Angular** and **Vue**.

In the following table we are going to display what is the current state of the world.

Metric	React	Vue	Svelte	Angular
Stars	176k	189k	50.7k	78k
Fork	35.4k	30.3k	2.4k	20.5k
Watch	6.7k	6.2k	874	3.1k
Contributors	401	23	445	1,498
Used By	7.6m	-	60.7	2.1m
Issues Open	885	539	618	1,847
Issues Closed	21,541	11,182	6,196	41,469
Existed Since	2013	2016	2016	2014

4 Approach

4.1 Data Fetching

For this paper we use Github API to get all the information from the projects, we get an API token, which increase the limits of their rate limit, we went through all the projects and we get all the commits and issues.

Commits: From commits we get the id, the message, the author and the date of the commit.

Issues: From issues we get the id, the message, the author, the date, the state (open or closed), and the comments, the comments are useful because this is what we are going to use for sentiment analysis

4.2 Sentiment Analysis

Each comments has a message and an author associate to it, we use a program that is able to provides us with a score about the sentiment of each comment, which could be either positive or negative, and also provides us a vector of words that can be either positive or negatives, which are used within the sentence.

4.3 Data Hosting

Once we iterate over this information we end up with two json files, one for issues and one for commits, which weights 405.7 MB and 16.2 MB respectively, which is an issue for sharing and also for processing, since this information is in a single file not all computers are able to allocate the entire file and memory and also make test slower.

For this reason we decide to upload this data into a MongoDB database that we can use, which is already optimized to handle this type of the structure, since this data is semi-structure, is in a JSON format and we needed to be able to filter, query or sort, MongoDB sounds like a good fit.

4.4 Edge Cases

For all the cases in which we are not able to get the data by using MongoDB query engine, we use scripting languages to fill these gaps, this scripts will connect to the MongoDB fetch the raw data and the script will do more complex queries with the data provided.

5 Experiment Design

There are two main things that we want to main questions that we want to answer to inform a potential contributor if a particular project would fit their goals.

5.1 Is the project alive?

The first one is to figure out if a particular project is alive, by **alive** we mean a project that it's in active development, it has a roadmap for the future and the community is active, there are examples of open source libraries that have an active community, the project still receive issues or pull request but from a roadmap perspective they already stablished that the project is done,

so no new functionality will be added, example of this is the library moment[2].

For this we are going to see how are the commits and issues behaving over time to see if there is any trend, either an increase or decline, that can help us as an indicator to see the "liveliness" of a project.

For the data that is going to be used we are going to group them by year, and group the projects together, and see what is the trendline for each of the metrics, these are:

1. Trend of commits over the years
2. Trend of opened issues over the years
3. Trend of unique contributors per year

5.2 What is the overall sentiment of the project?

The second one is related to the community behind this projects, we want to know if they are positive, negatives or neutral, and into which degree.

For this we are going to use the comments inside the issues, the reason we are not using commit messages is because this one are related to the code, but comments in an issue have a more social nature, so comments would be a better indicator of the project "personality" that the commit message.

We fetch data from all the issues and each issue has a comment property that includes multiple messages send by the contributor, collaborator or any interested party, for each of this comments we performed a sentiment analysis on which we segmented the comment into words and group the positive and negatives words, then we see a comparative value between this and assign an score, if the word is positive we assign a positive score and if it's negative we assign a negative value, we also keep a list of all the positive and negative words for future analysis.

6 Results

6.1 Initial Analysis

We discovered that the front-end languages react and angular received a disproportionate number of commits during the initial analysis. These data show a distinct trend in language usage over time, with the number of commits for react decreasing from the year 2015, which was 2656, to the year 2021, which was 778, and the same is valid for angular. In 2015, there were 3364 commits, and by 2021, that number will have dropped to 2278.

Compared to other front end languages, these two languages received the most significant number of commits because they were both the most ancient languages. During those years, most front-end applications relied solely on these programming languages to achieve their functionality.

The vue language had the lowest number of commits compared to all other languages. This is because it is a new developing language and people are more familiar with the front end languages they use from the beginning.

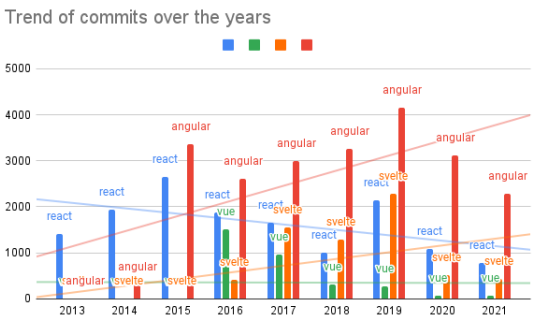
According to the data in all of the graphs, the number of open issues for Angular was the highest compared to all other languages, and the same was true for the number of closed issues. When it comes to frontend languages, Angular is most commonly used, and a wide range of applications supports it. Despite this, the number of open issues for new languages has increased in the years following 2019. We can assume that people started working on these languages because they offer better scalability, performance, security, and other advantages over react and angular.

6.2 Answers

6.2.1 Is the project alive?

We are going to use **time** as the basis to which we are going to attach the metrics, if we see an increase overtime around a project we could say that a project has an active community and the contributions, which could be issues or commits, are increasing over time, otherwise if we see a decline we can see how steep this decline is, and raise that so the contributor take that into consideration.

Trend of commits over the years: We use the following data set [3] that count all the commits that were performed for each project each year and we plot it in the chart below.



We can see that for react an increase in the first 3 years but after that there has been a decline in the amount of commits, on the other hand we can see that angular started one year later after react and it surpass it in 2015, and the rate of commits has been increasing ever since.

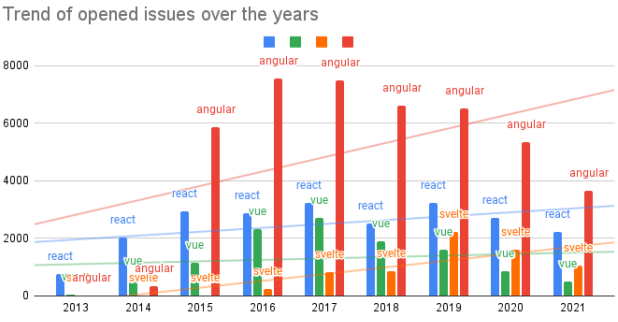
Vue and Svelte are the newest framework, both conceive in 2016, vue's contribution has been more stable over the last few year while svelte has a trend of increasing, and the amount of commits seems to surpassed vue most years.

Even tough react is the only project that has a declining trendline we don't think that is dying, if we take a look at the table provided in the section *Current state of the world*, react is used by 7.6 million projects.

We can't take react outside of the context on which it was born, inside Facebook, to standarize the way that developers create user interfaces, we attribute this decline to the maturity of the framework, using the Capability maturity model [4], tell us that the project is at level 5, and their focus right now is to improve performance while keeping the stability of the project.

Trend of opened issues over the years: The last trend tell us how developers interact with the project, while this trend emphasizes how the community interacts with the project, issues are the social component in open source, at least in github.

In the following chart we did the same approach as before, we take all the date where the issues were open and we group them by year, and we put the projects next to each other to get any insights, below we have a chart that depict a summary of the issues over the years, for this we use the following data set [5].

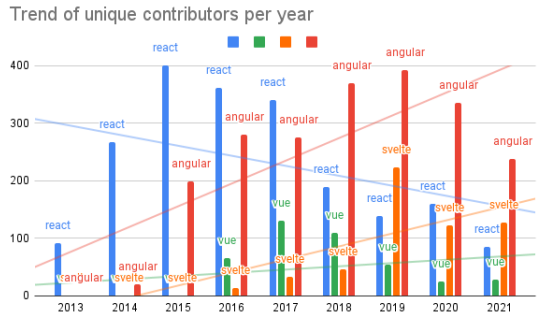


From the following chart we can see that angular has the most active community overall, from react we can see that we have an increasing trendline over the years, which contrast with the decline that it's perceive from the previous trend, which

helps with the assumption that we make before that commit's alone are not a single representative metric that provide *liveliness* to a project, vue has a slightly increase over the years and has the same trend behaviour that was displayed in the previous trend, svelte seems to have a increase trendline as well, surpassing vue community in the last 3 years.

This could tell us that overall, all of this communities are fairly active which suggest that the project is alive, from an social aspect.

Trend of unique contributors per year
We already saw the amount of contributions a code received, in the form of commits, and how the community interact with each other, now are we going to see how big the community is by the amount of unique developers that contribute to the project, we are going to use the following data set[6] to create the chart that is shown below.

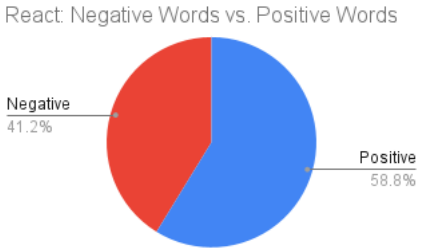


From the chart we can see a decline in react from 2015 to 2021, it could be because at the beggining more developers were joining but with the pass of time the community was getting more mature and the contributions got centralized which coincide with the trend of commits over the years, angular has receive an increase in unique contributors each years which is a sign that the amount of developers is growing, svelte has seen an increase in the last years and vue has receive a fairly increase of contributors since it's inception.

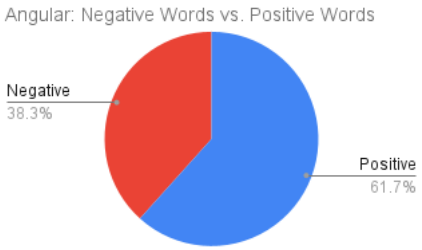
6.2.2 What is the overall sentiment of the project?

Now we are going to evaluate what are the positive and negative sentiments for each project, using the following data [7].

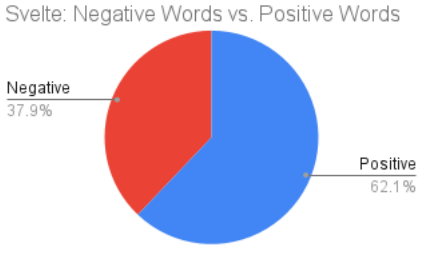
Below we are going to see a chart that depict what is the percentage of positive and negative words used in each projects.



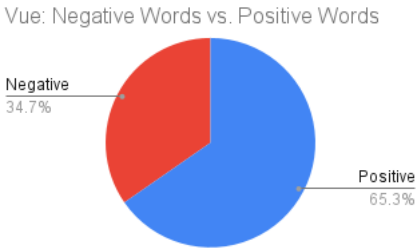
In this chart we can see that react has more positive (162,415) words that represents 58.8% compare to the negative (113,892) words, which represents 41.2%.



In the case of angular we have a more positive community than react with 280,779 positive words, which represents a 61.7%, compare with the 174,500 negative words which represents the 38.3%.



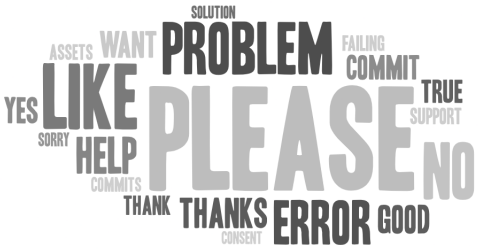
Next we have svelte with 35,965 positive words that represents the 62.1% and 21,948 negative words that represents the 37.9%.



Vue has a total of 44,901 positive words, 65.3%, and 23,821 negative words, 34.7%.

This informations suggest that all the communities are positive in general, and there are big differences between each other, now we are going to display what are the most used words, top 20, using cloud words and for each project we are going to see if it gives us an indication of what the community looks like.

This cloud words is for **react**, from this we can see that the most used word is **like**, we also encounter negative words like error, warning and danger but in a



This cloud words is for **angular**,

This cloud words is for **vue**,

This cloud words is for **svelte**,

7 Conclusion

References

[1] Jason Warner. Thank you for 100 million repositories. https://github.blog/2018-11-08-100m-repos, accessed 2021-12-08.

[2] Moment.js. Parse, validate, manipulate, and display dates and times in JavaScript. https://momentjs.com, accessed 2021-12-08.

[3] Jose J. Cruz. Trend of commits over the years. https://github.com/jjzcru/stevens-533/blob/main/final/data/trend-of-commits-over-the-years.csv, accessed 2021-12-09.

[4] 02DCE. Software Engineering — Capability maturity model (CMM). https://www.geeksforgeeks.org/software-engineering-capability-maturity-model-cmm/, accessed 2021-12-09.

[5] Jose J. Cruz. Trend of issues over the years. https://github.com/jjzcru/stevens-533/blob/main/final/data/trend-of-open-issues-over-the-years.csv, accessed 2021-12-10.

[6] Jose J. Cruz. Trend of unique contributors per year. https://github.com/jjzcru/stevens-533/blob/main/final/data/trend-of-unique-contributors-per-year.csv, accessed 2021-12-10.

[7] Jose J. Cruz. Overall sentiment. https://github.com/jjzcru/stevens-533/blob/main/final/data/overall-sentiment.csv, accessed 2021-12-10.