First time contributors guide for front-end frameworks

Team 7

Akshay Prasanna Kunal Satija Jose Cruz

Stevens Institute of Technology

December 13, 2021

1 Abstract

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.

Because front-end developers are in high demand, this paper will look at what sets one project apart from another.

We hope to shed light on the most popular frameworks (react, vue, angular, and svelte) and their communities by publishing this paper.

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

2 Introduction

We want to focus on the front-end because it is a simple way for a new contributor to get involved, despite many open-source projects available on Github (over 100 million projects[1]). Even more importantly, the collaboration will have a significant impact on the lives of several developers and, as a result, their customers.

We will focus on the four most popular front-end frameworks, as well as the projects and communities built around them throughout this paper, to help potential first-time contributors decide which framework to use.

3 Background and Motivation

3.1 Background

When a developer wishes to start its journey as a contributor, they don't know how to begin, we suggest choosing one of the suggested front-end frameworks, it will not only increase the developer's skills but will also help with the interaction with other contributors.

This kind of interaction provides, give first-time contributors access to developers with far more experience which in turn will help them to find more opportunities outside of the scope 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 increases the limits of their rate limit, we went through all the projects and we get all the commits and issues, and stores them in JSON files.

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

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

4.2 Sentiment Analysis

We read and parse all the issues from the JSON file, each comment inside the issue has a message and an author associate with it, we split the content into words, remove all the special characters and we categorize each word into positive and negative, and with that, we assign a score to the comment.

4.3 Data Hosting

The two JSON files generated in the **Data Fetching** section are large, the *issues* file weighing in at 405.7 MB and the *commits* file weighing in at 16.2 MB, making the process of querying data slow because we would need to mount all the data into memory for each request, as well as create a script for each type of query we would like to perform on the data.

To avoid this, we will host that information in a MongoDB instance; because the data is already in JSON, no custom transformation is required, and they have a query language that supports JSON, making it easier and faster to run queries against the data.

4.4 Edge Cases

We use scripting languages to fill in the gaps where the MongoDB query engine cannot obtain the data. These scripts connect to the MongoDB instance, retrieve the JSON, and then perform more complex queries with the data provided.

5 Experiment Design

There are two main things we want to know about the projects, to see if they are a good fit for a first-time contributor.

5.1 Is the project alive?

It is essential to determine whether or not the project is still being developed and maintained; by **alive**, we mean that there are still issues or pull requests being submitted to the project, but the project's roadmap has already established that no new functionality will be added; the library moment [2] is an example of this.

So, as an indicator of how active a project is, we will look at how commits and issues

change over time; we will use this as a measure of its "liveliness."

For this analysis, we will group the data by year and then examine the trends for each of the following metrics:

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?

Second, we would like to know whether the people working on these projects are positive, negative, or neutral, and to what extent.

Rather than using commit messages, which are more closely associated with the code, we will look to the comments within issues to get a sense of the project's "personality", because comments are more social.

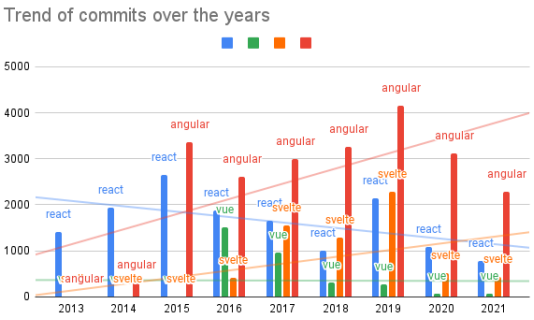
There are multiple comments on each issue, and we perform a sentiment analysis on each of these comments; for each of these comments, we group the words into positive, neutral, and negative, then we see the comparative value between this and assign a score. If the word is positive, we assign a positive score; there are multiple comments for each word, and each comment has a sentiment analysis. If it is negative, we assign a negative value; we also keep a list of all positive and negative words for future reference.

5.3 Answers

5.3.1 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 over time 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 takes 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.



When it comes to React, the number of commits increased for the first three years of its existence but has since declined, whereas the rate of commits for Angular has been increasing since its inception in 2015.

Vue and Svelte are the two newest frameworks as of 2016, and while Vue's contribution has been more stable in recent years, Svelte appears to outpace Vue regularly.

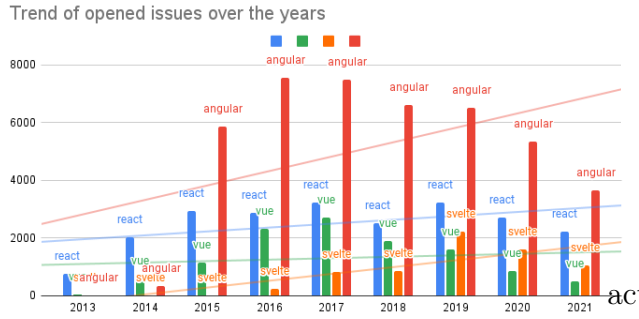
Even though React is the only project that has a declining trendline we don't think that is dying, if we take a look at the table provided in the section *Current state of the world*, React is used by 7.6 million projects.

We can't take react outside of the context in which it was born, inside Facebook, to standardize 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:

While the previous trend emphasizes how the community interacts with the project, issues are the social component in open source, at least on GitHub.

In the chart below, we used the same approach as before: we took all the data where the issues were open, grouped them by year, and put the projects next to each other to get any insights; below, we have a chart that depicts a summary of the issues over the years, and we used the following data set [5].

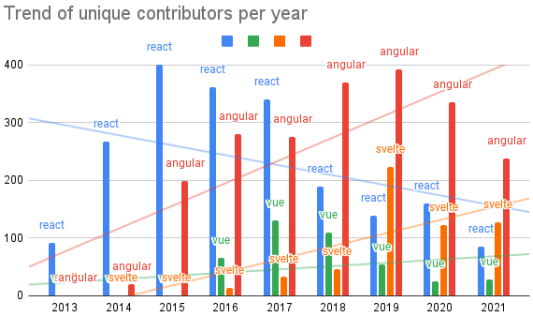


From the chart we can see that angular has the most active community overall, from react we can see that we have an increasing trendline over the years, which contrast with the decline that it's perceived from the previous trend, which helps with the assumption that we make before that commit's alone are not a single representative metric that provides *liveliness* to a project, Vue has a slight increase over the years and has the same trend behavior that was displayed in the previous trend, svelte seems to have an increase trendline as well, surpassing Vue community in the last 3 years.

This could indicate that, in general, all of these communities are relatively active, implying that the project is socially active.

Trend of unique contributors per year

We have already seen how many contributions a project received in the form of commits and how the community interacts with one another, now we will see how big the community is by the number of unique developers who contribute to the project, and we will use the following data set [6] to create the chart below.

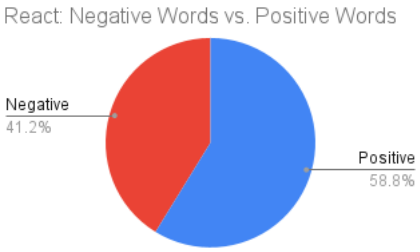


From the chart we can see a decline in React from 2015 to 2021, it could be because at the beginning more developers were joining but with the pass of time the community was getting more mature and the contributions got centralized which coincide with the trend of commits over the years, angular has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

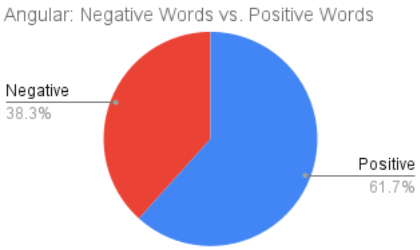
5.3.2 What is the overall sentiment of the project?

Using the following data [7], we will now assess each project's positive and negative sentiments.

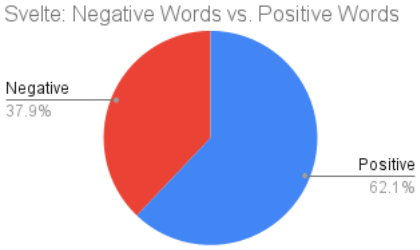
We will see a chart below showing the percentage of positive and negative words used in each project.



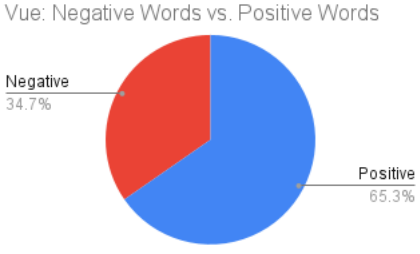
In this chart we can see that react has more positive (162,415) words that represent 58.8% compared to the negative (113,892) words, which represent 41.2%.



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



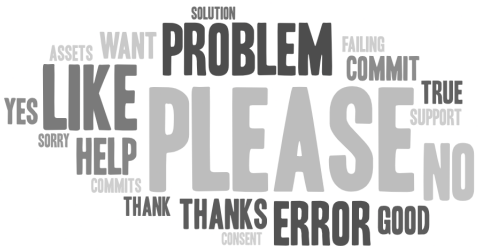
Next, we have svelte with 35,965 positive words that represent 62.1% and 21,948 negative words that represent 37.9%.



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

This data suggests that all communities are positive in general, but there are significant differences between them. We will now display the most frequently used words, the top 20, using word clouds, and for each project, we will see if it gives us an indication of what the community looks like.

React



Angular

Vue

Svelte

From the word clouds we can see that the most use words is **Like** and **Please**, and most uses get used accross multiple projects like **thanks**, **thank**, **good**, **no**, **problem**, which suggest that there is not a specific words that difference one community from another and overall all the communities are positive, because if we take a look at the negative words are words like **no**, **danger** or **problem** which could for fixing a particular problem.

6 Analysis

Compared to svelte and Vue, **React** and **Angular** have made the most significant overall contribution because they were the first while being backed by large companies React (Facebook) and Angular (Google), which drove early adoption.

In general, all the projects have an active community and the projects are being developed, and the community seems to use the same kind of word, so there isn't any bad choice from a community side.

If we are talking about impact, we would recommend Vue because it has the fewest contributors, making it easier to leave your mark on the project. However, if you want to learn new skills and have a larger community, we recommend Angular and React, which have the most contributors.

If the contributor wants to learn something new and take a different approach than the other frameworks, we recommend Svelte, but the fourth project is excellent for the metrics we establish, the "liveliness," and their communities in general.

7 Future Work

Even though this paper focused on the most popular front-end frameworks, the approach for the community could be expanded by including Reddit or StackOverflow.

Since this one is primarily social it could give us a view of how the community interacts outside of the scope of the project, while having the usage of the project as the main topic.

The method described in this paper could be applied to a wide variety of projects and libraries, not just for first-time contributors but also for seasoned contributors who want to contribute to a project in a specific area and have various options.

8 Conclusion

Open source may appear daunting at first, but working on it will help developers improve their skills and build a support network, each open source community is unique. Each group has its own distinct set of members, each with responsibilities and benefits.

Angular, Vue and Svelte have been increasing in their commits and community, while React has been declining but this would be a sign of maturity rather than a dying project, after all, is back by Facebook and it's an integral part of their ecosystem.

We also layout a potential use case for the project and how the community analysis could be improved to better depict how the communities interact outside of GitHub.

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.