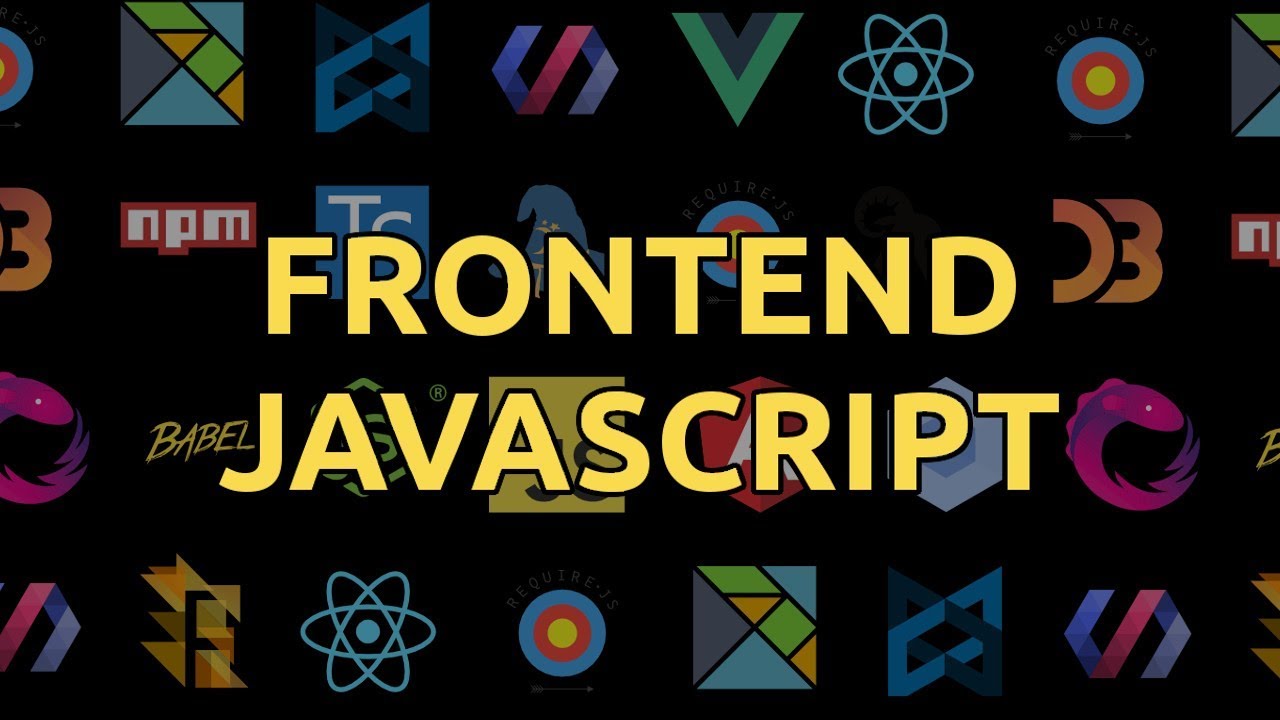
Front-End Research



Jeroen Koeter

S3-DB01

Inhoud

[Vue 3](#_Toc67514040)

[Angular 4](#_Toc67514041)

[React 6](#_Toc67514042)

[List of sources: 7](#_Toc67514043)

# vue logoVue

Vue is an open-source JavaScript framework. It is easy to learn and very versatile, this is because Vue is a sort of “plug and play” framework. You can use Vue as a basic easy framework with not a lot of functions, but you can also plug new components into Vue to expand it. Some of the components are a router or state management for example. Vue is made by a community and not sponsored by a mega corporation like Angular (Google) or Facebook (React). This means that Vue is completely free in the way it is developed and is not pushed in a certain direction by its sponsors.

Pros of Vue:

* Vue utilizes virtual DOM rendering which will boost performance. This means that Vue will “copy” the DOM of the loaded page and looks at what needs to be updated with each user interaction then loads this virtually, this means that not the entire “real” DOM needs to be updated but only the changed parts.
* Vue has reactive two way binding. Two way binding is a connection between data and view, this means that bound components have data stored which can be updated. In Vue this bound data is updated reactively which means you can update your webpage in real time.
* Vue uses single file components, this is different from the way Angular and React approach components. In Vue all your components are in a single file which greatly improves the readability. Splitting your code in components is a architectural design called Component Based Architecture or CBA for short. CBA helps greatly with unit testing and code reusability.
* Vue is very easy to learn, to start with Vue you only need to know HTML, CSS and JavaScript. You don’t need to know TypeScript or have any in depth knowledge on libraries.
* Vue has a lot of documentation and also a lot of community support. When you have a problem in Vue the answer is easy to find a lot of the time to the great documentation and its really active community.

Cons of Vue:

* There can be a language barrier with learning Vue, this is the case because Vue was popularized by companies like Xiaomi and Alibaba. The problem with this is that you’ll most likely also find a lot of its documentation is in Chinese. This can be a problem for non-Chinese speaking programmers.
* The reactive two way binding is complex to get to work, this is because the way it works isn’t smart and the system can make mistakes while reading data. This is a well know issue with Vue and it’s also addressed in a lot of its documentation.
* Vue has limited resources and a lack of experienced developers, both these cons are due to the fact that Vue is still in its infancy period. It’s a quite new framework and both these issues will disappear over time when it will be more widely used.

# Angular - WikipediaAngular

Angular has been around for a long time, it started its life in 2009 as the framework currently know as AngularJS . The first official release was in 2010. It’s inventors Misko Hevery and Adam Abrons were both Google engineers. Angular is also supported by Google. In 2016 Angular 2 was released, this was such a big upgrade from AngularJS that is wasn’t just an update anymore. Angular 2 was a complete rewrite of AngularJS. Angular 2 has been had numerous updates since then, the Angular we now use is actually Angular 9. At its core Angular is an typescript based open-source framework led by the Angular team at Google.

Pros of Angular:

* Angular has a component base architecture (CBA). Because it uses a component base architecture it is very readable which is very nice if you work in a bigger team or if someone needs to work on code not written by themselves. The CBA also means that it is easier to write Unit tests for your application, this because every component is independent. Your code will also be more maintainable thanks to CBA, this is simply because ever component is independent so you can just swap out components for other components if necessary. Lastly CBA also ensures that your code is reusable, this is because components will be encapsulated which means that you can use them for different parts of the application.
* Angular uses TypeScript. TypeScript is a better version of JavaScript. TypeScript has better tooling such as navigation, autocompletion and refactoring services. It also catches a lot of common mistakes while you type your code. TypeScript also completely compiles to JavaScript.
* Angular also works really well with the RxJS library. The RxJS library is a library for asynchronous data calls, this means that your events are handled in continuing parallel execution. This means that you webpage doesn’t need to wait for event 1 to end to start event 2. As Thinkster.io puts it: **“RxJS is to Javascript as Henry Ford's assembly line was to cars.”**
* Long term Google Support, this means that Angular will be kept up to date and, at least in the foreseeable future, won’t be outdated. The other great thing about the long term support Google promised is that this means that they’ll keep trying to stay in front of other frameworks, so there’ll probably be a lot of new function and even better optimization in the future.
* Angular uses directives. Directives have been around since the AngularJS days, the great thing is that with the transition to Angular back in 2016 directives were also ported to the component based architecture. With directives you can extend HTML capabilities to change DOM trees.

Cons of Angular:

* Angular has a massive community, this might not seem like a bad thing at first sight. The problem is that the community is very divided, this is because the transition from AngularJS to Angular was a hard one. AngularJS support is to be stopped June 2021 so a great part of the community will likely disappear after this date.
* Angular has a very steep learning curve, this is because there is an entire list that you’ll need to learn to be able to use Angular: Components, Services, Templates, Dependency Injection, RxJS, Modules and the list goes on. So a lot to get into.

# Bootstrapping a React projectReact

React is a library that feels like a framework. React takes advantage of the speed of JavaScript and a new way of rendering webpages which makes the pages very dynamic and also really responsive to user input. React has been developed by Facebook to overcome major UX tasks like building a high performance dynamic UI. In 2013 React was released as an open-source JavaScript tool. Two years later React Native was released, this is an “Hybrid mobile-app development” framework for IOS and Android. React native was based on ReactJS.

Pros of React:

* React uses a virtual DOM, the use of an virtual DOM makes the webpage adapt quickly to input. A virtual DOM also makes it so pages don’t need to be reloaded in its entirety every time the smallest of details change. This is because React loads these changes in the virtual DOM and compares is to the real DOM, after this comparison only the changed items are reloaded.

* React uses an Component Based Architecture (CBA), this means that each component is independent. This makes writing unit test a lot easier, furthermore it results in high readability of your code. Your code will also be easier to maintain. Another advantage of CBA is that you can easily reuse your components, this is because they are completely independent.
* React uses an one-directional Dataflow. This will result in more stable code, because child elements can no longer change parent elements. In an view-model system child elements may change the parent element, but Facebook fixed this by making React a view system.
* React is at its core an open-source Facebook library, which results in a lot of people working on it. The great benefit of this is that there are a lot of people working on this library, this also means that there are an insane amount of extra tools created by the community.

Cons of React:

* Because of the way React works it is updated a lot, this means that you’ll need to relearn ways to do things quite a lot. This is something that can put you off of using React.
* React also has poor documentation thanks to its ever evolving existence. There is a lot to be found but most of the documentation is from developers writing their own documentation while using the tools.
* JSX is also a con of React in the eye of many, this is because it allows you to mix your HTML and JavaScript. This can make your code sloppy and look like spaghetti code. JSX also has a quite steep learning curve.

# List of sources:

* <https://www.youtube.com/results?search_query=react+vs+angular+vs+vue>
* <https://www.youtube.com/watch?v=Tn6-PIqc4UM>
* <https://www.youtube.com/watch?v=nhBVL41-_Cw>
* <https://www.youtube.com/watch?v=Ata9cSC2WpM>
* <https://www.freecodecamp.org/news/the-difference-between-a-framework-and-a-library-bd133054023f/>
* <https://www.altexsoft.com/blog/engineering/pros-and-cons-of-vue-js/>
* <https://ddi-dev.com/blog/programming/the-good-and-the-bad-of-vue-js-framework-programming/>
* <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/>
* <https://pagepro.co/blog/react-vs-angular-a-quick-comparison/>
* <https://thinkster.io/tutorials/learn-rxjs-observables/what-is-rxjs>
* <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-reactjs-and-react-native/>