Comparative study: Angular 2 vs React

# What is ReactJS?

ReactJS is an open-source front-end library which focuses on the V(iew) in the traditional MVC architecture and was manufactured in 2013 by Facebook. Like Angular 2, React is heavily focused on components. It uses a special syntax called JSX which basically is simple JavaScript mixed with HTML quoting (like XML) which provides fast rendering and compiles to plain JavaScript.

ReactJS has a single-way data flow which means all data flows down from the root component to its subcomponents and all events flow upwards but this can made unidirectional with the Redux library. React also uses the Virtual DOM which is stored in-memory to improve performance on the front-end.

# Comparing Angular 2 and React

Syntax

Considering Angular 2 is pretty much a fully-fledged framework compared to the smaller React which focuses more on the UI, it is safe to say that JSX is most useful for React because of the HTML templating.

Since React is far older than Angular 2 and Typescript has only gained a lot of popularity the past two years or so this might be an explanation because of why React isn’t written in Typescript. Although React doesn’t out-of-the-box come with a type safety library or package it does have the PropTypes package which can provide some type safety for props. It’s also possible to get the Flow library which works with React, JSX and Redux and provides these type safety features that Typescript also offers. Nevertheless, Typescript is now supporting JSX syntax so it would technically be possible to mix both Typescript and JSX into one.

Structure

Since both React and Angular 2 are heavily oriented around components they do have significant similarities in the app structure. Both start from a root html file to load in the main root component and components are ordered via a components folder map or just the component name as a folder name. There’s an assets folder in both as per usual. Besides that it does seem more common to split up the logic a lot more in Ng2 because of it’s OOP/MVC background such as module/service/pipe/… files and/or folders depending on the user’s needs. Obviously both frameworks are flexible and there’s a lot of boilerplates out there that will likely satisfy the user.

Building the App

Both React and Ng2 have great CLI and GitHub support for a project setup/installation.

The two both require the same initial setup where you a main module file for Ng2 and an index file for React in which the routing must be setup and all the components must be loaded in and for React Redux initialization. Ng2 is a bit more tedious in this regard since you need to include all Angular modules and all the other files you created yourself which you will use in the main module file.

Past this point setting up components is for both really flexible and is handled in a very similar way: you import everything you need at the top of the page and can freely build up your component from that point.

Data-binding

As mentioned before React out-of-the-box has one-way binding, but generally developers add the Redux library when needed which provides central state management to make the data-binding unidirectional such as is present in Angular 2.

DOM Differences

React uses the virtual DOM which is basically a copy of the regular DOM and is quite useful to rapidly detect ‘dirty’ sections for performant re-rendering of the regular DOM. Angular 2 uses the regular DOM and therefore sometimes suffers in performance.

Difficulty

When looking at both React and Ng2, it seems rather clear that picking up React is far easier than picking up Ng2, simply because React is a library and Ng2 is a big framework. React doesn’t have all the modules that Ng2 has such as an HTTP module or a Forms module. Furthermore, React’s syntax is really easy to learn since you don’t need to get accustomed to Typescript, which takes a while to get used to because of all the type safety shenanigans.

All those arguments entirely depend on a user’s background and needs. If you like React’s simplicity and flexibility whereas you can add libraries you prefer or have worked with before then React would be best suited for you. But if you wouldn’t want to bother with installing a lot of additional libraries to satisfy your needs than Angular 2 might be a good option for you if you want to sit down for a few days learning about all the Ng2 modules as opposed to picking up React on the go in a couple hours (Redux excluded).

Performance

A study isn’t complete without some performance testing, to compare React and Angular 2 I chose to make use of using 10.000 rows where I will measure performance whilst executing a few simple tasks like making 10.000 row, removing them, clearing some rows, amending some and so on and will afterwards display the loading time results. These tests were performed with both the latest version of React and Angular 2. All amendments were done by creating 10.000 rows first.

|  |  |  |
| --- | --- | --- |
| Average number of 10 attempts taken in milliseconds (ms) | React | Angular 2 |
| Make 1000 rows | 199,7 | 208,8 |
| Make 10.000 rows | 2.127,2 | 2.433,6 |
| Append 1000 rows | 254,2 | 259,4 |
| Update every 10th | 159,9 | 129,2 |
| Clear all rows | 361,5 | 287,2 |
| Swap random rows | 175,6 | 176,5 |
| Delete one row | 561,4 | 601,5 |
| Select row and colour it | 36,6 | 12,1 |

# Sources

<https://www.cleveroad.com/blog/react-vs-angular-ultimate-performance-research-2017>

<https://artjoker.net/blog/angularjs-vs-reactjs/>

<https://discuss.reactjs.org/t/if-typescript-is-so-great-how-come-all-notable-reactjs-projects-use-babel/4887>

<https://medium.com/javascript-scene/angular-2-vs-react-the-ultimate-dance-off-60e7dfbc379c>

<https://gorillalogic.com/blog/angular-2-vs-react/>

# Logs

**06-08**

* Making a few basic React applications to showcase how React works split up into state, connecting an API and redux
* What is ReactJS?

**07-08**

* Comparing syntax, JSX vs Typescript
* Comparing common folder structures of Ng2 and React
* Comparing the build setup of both frameworks
* Data-binding, the differences
* Virtual and regular DOM
* How difficult are both to learn
* Performance testing both framework with the 10.000 rows test
* Displaying the performance test results in a table