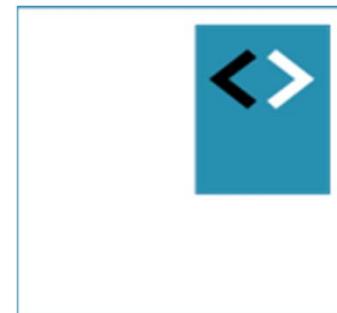
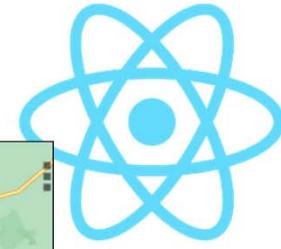


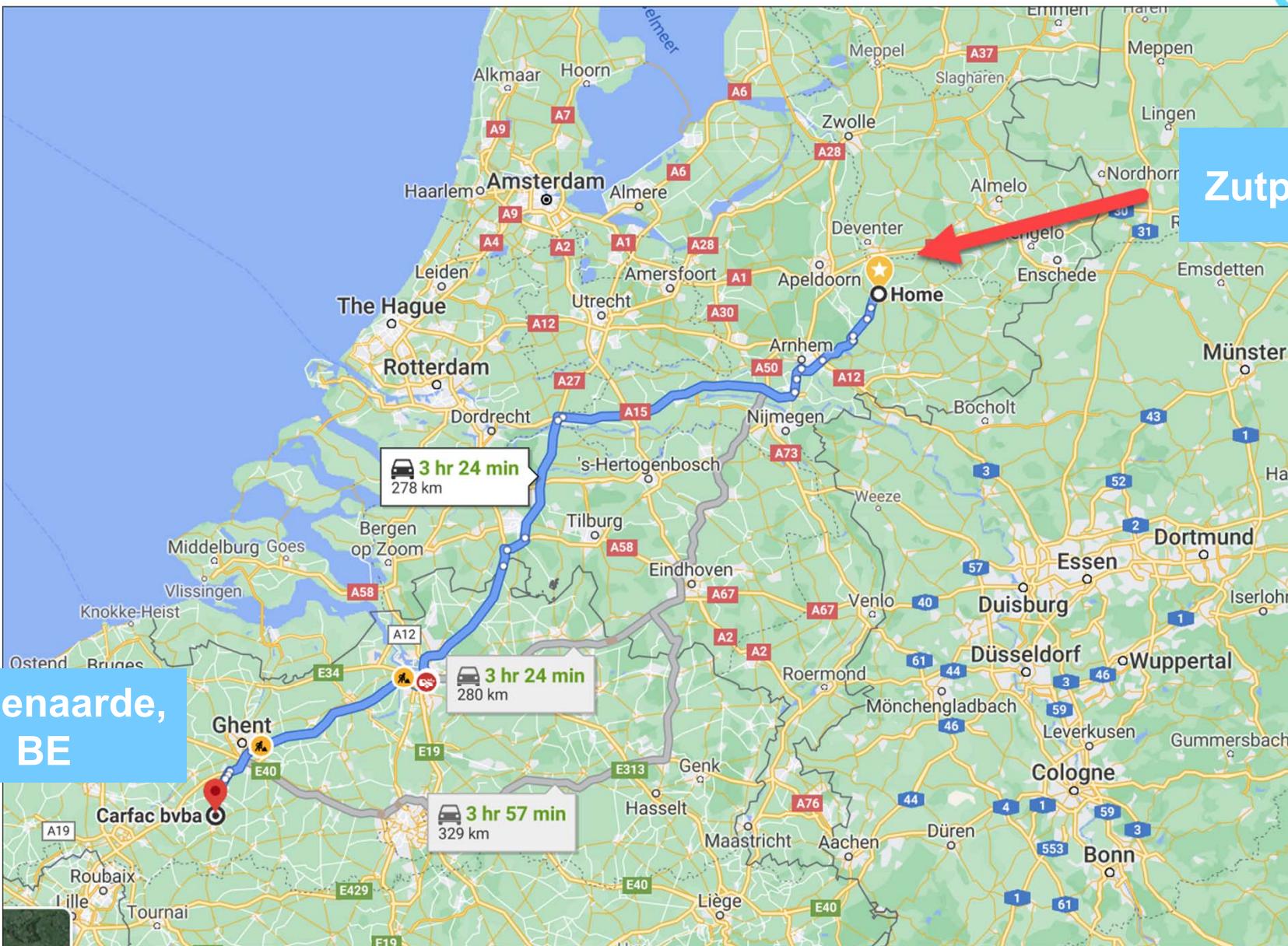
# React Fundamentals Module - Introduction



Peter Kassenaar –  
[info@kassenaar.com](mailto:info@kassenaar.com)



Zutphen, NL



Oudenaarde,  
BE

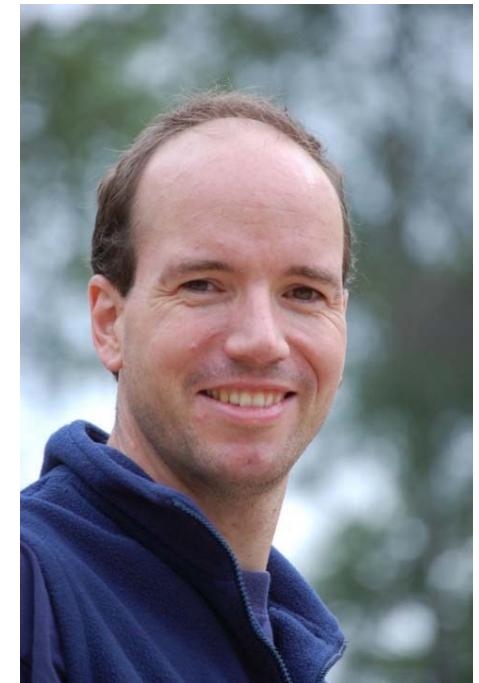
# Peter Kassenaar

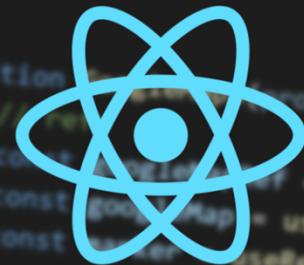
- Trainer, author, developer – since 1996
- Specialty: “*Everything JavaScript*”
- JavaScript, ES6, Angular, NodeJS, TypeScript, jQuery, React, Vue, etc.

[www.kassenaar.com](http://www.kassenaar.com)

[info@kassenaar.com](mailto:info@kassenaar.com)

Twitter: [@PeterKassenaar](https://twitter.com/@PeterKassenaar)





# Reacttraining.nl

World-class React training in Dutch and English

[Learn more](#)

We teach React

[www.reacttraining.nl](http://www.reacttraining.nl)

<https://github.com/PeterKassenaar/carfac>

The screenshot shows the GitHub repository page for `carfac`. The repository was created by `PeterKassenaar` and has 1 commit, 1 branch, and 0 tags. The repository description is: "Slides and example code on the training React Fundamentals, Carfac, November 2020". The repository has 1 star, 0 forks, and 0 issues. The repository page includes sections for Octree, Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings.

**Code**

Octree > PeterKassenaar Initial commit b117cab 42 minutes ago 1 commits

- .gitignore Initial commit 42 minutes ago
- LICENSE Initial commit 42 minutes ago
- README.md Initial commit 42 minutes ago

Go to file Add file ▾ Code ▾

**About**

Slides and example code on the training React Fundamentals, Carfac, November 2020

Readme MIT License

**Releases**

No releases published Create a new release

**Packages**

No packages published Publish your first package

[github.com/PeterKassenaar/react-fundamentals](https://github.com/PeterKassenaar/react-fundamentals)

The screenshot shows the GitHub repository page for `PeterKassenaar / react-fundamentals`. The repository has 38 commits, 2 branches, 0 packages, 0 releases, 1 contributor, and is licensed under MIT. The README.md file contains the text "react-fundamentals".

Example code and exercises on the training React Fundamentals by Peter Kassenaar

react training webdevelopment webapplication Manage topics

38 commits 2 branches 0 packages 0 releases 1 contributor MIT

Branch: master New pull request Create new file Upload files Find file Clone or download

PeterKassenaar	Update README.md	Latest commit ee6158a 19 days ago
examples	Added example 720-custom-hooks	3 months ago
training-app	updated training app	2 months ago
workshops	Added example 140-class-components	3 months ago
.gitignore	Added 100-hello-world	3 months ago
JavaScript APIs.txt	Added JavaScript APIs.txt	3 months ago
LICENSE	Initial commit	4 months ago
README.md	Update README.md	19 days ago

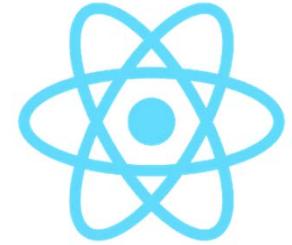
README.md

react-fundamentals

# About you...



# Tell us a little bit about yourself



Knowledge of React, other front-end frameworks?

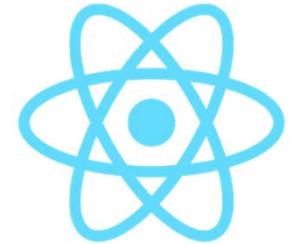
Have you worked with HTML, CSS, JavaScript,

TypeScript and/or other (web)languages?

Tell us a little bit about your projects.

What are your expectations of this course?

# Material

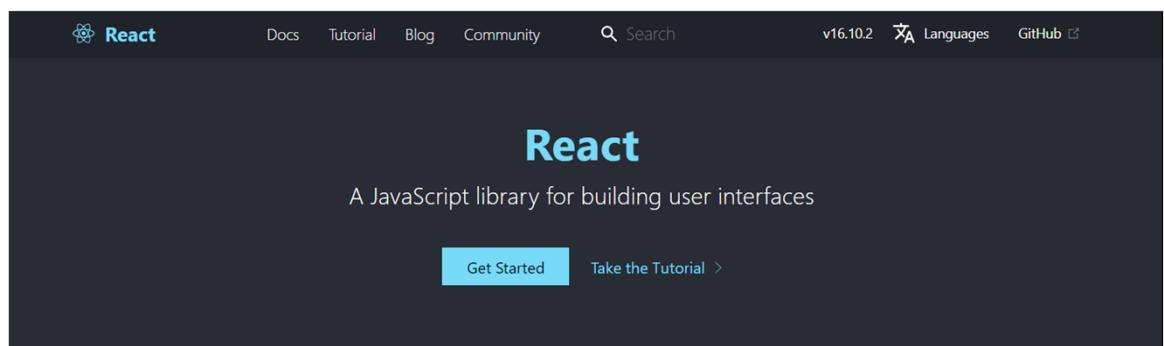


Software (React + libraries)

Handouts (PDF, Github)

Workshops (In Presentations)

Websites (online)

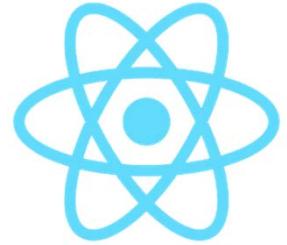


The screenshot shows the official React.js website. At the top, there's a dark header bar with the React logo, navigation links for Docs, Tutorial, Blog, and Community, a search bar, and version information (v16.10.2). Below the header, the word "React" is prominently displayed in large blue letters, followed by the subtitle "A JavaScript library for building user interfaces". There are two calls-to-action: "Get Started" and "Take the Tutorial >". The main content area is divided into three columns: "Declarative" (React makes it painless to create interactive UIs), "Component-Based" (Build encapsulated components that manage their own state), and "Learn Once, Write Anywhere" (We don't make assumptions about the rest of your technology stack). At the bottom right, there's a red link to "reactjs.org/".

Declarative	Component-Based	Learn Once, Write Anywhere
React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the	Build encapsulated components that manage their own state, then compose them to make complex UIs.	We don't make assumptions about the rest of your technology stack, so you can develop new features in React without rewriting existing code.

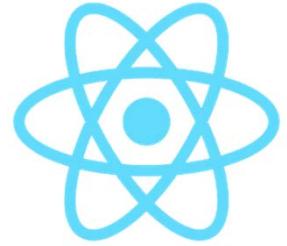
[reactjs.org/](https://reactjs.org/)

# Agenda - 3 days



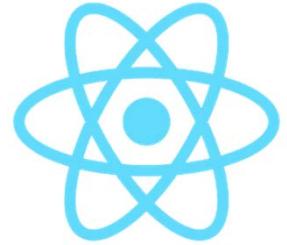
- Introduction – overview of the front-end landscape
- React tooling – installation
- Hello World –the structure and architecture of React apps.
- Zooming in:
  - Components – Class based vs. Function based

# Agenda – cont'd



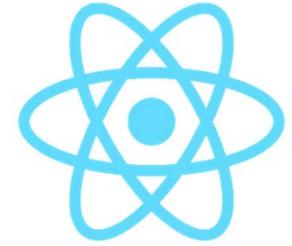
- React hooks - introduction
- Data and state in components
- Styling React components
- Communication between components
  - Props and events
- Lifecycle Hooks
- Handling user input

# Agenda – cont'd



- Communicating with **external API's**
- Using **React Router**
  - Alternative – Reach Router
- **React Forms**
- **React hooks** in depth
  - useState, useEffect, custom hooks
- Intro in managing state with **Redux**
- **Deploy** your app to production

# Labs and example code



## 1. Labs/Exercises

- In the PDF's in the Github-repo. But: feel free to deviate. Adapt to suit your own needs! (hobby, work, current projects)

## 2. Example code

- Executions of the exercises, small projects (`npm install`, `npm run serve`)
- Work in progress – let me know of additions/errors!
- [github.com/PeterKassenaar/react-fundamentals](https://github.com/PeterKassenaar/react-fundamentals)

# Global Agenda



25-26-27 Nov. 2020 – **Wed–Thu–Fri**

~09:00 start

~ 10:30 coffee break

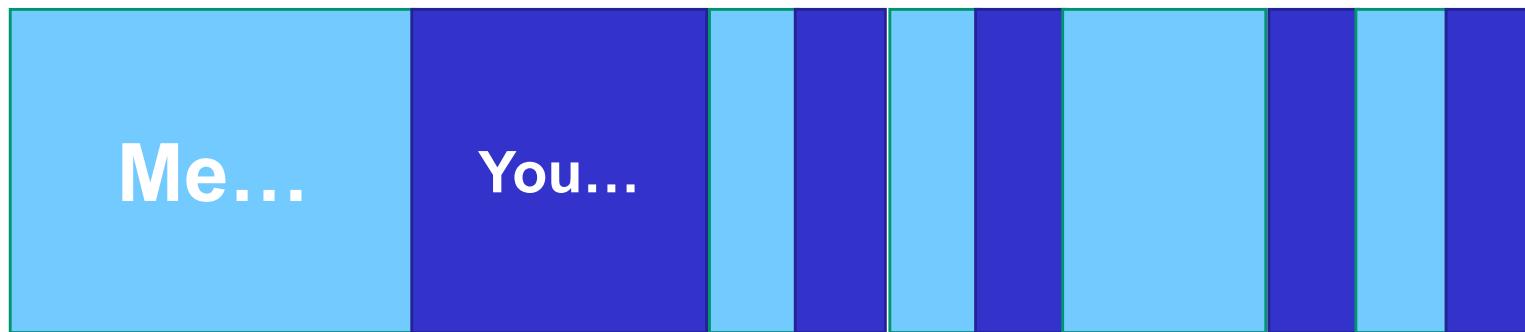
~ 12:00-13:00 Lunch Break

~ 14:30 coffee break

~ 16:15-16:30 wrap-up

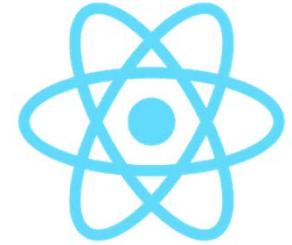
Friday – probably wrap-up a little bit early

# Overall process



# Questions?

# Addressing the “WHY” question!



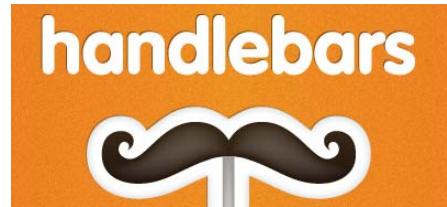
WHY, would we want to use a frontend framework.

It is all **HTML**, **CSS** and **JavaScript** right? speed,  
not re-inventing  
the wheel,  
consistency,,  
community,  
performance,  
testing....

Rethorical question:  
**“Do we want to go back  
to the jQuery days?”**

# Old school web apps

HTML + templates



Data Binding



Routing



DOM-manipulation

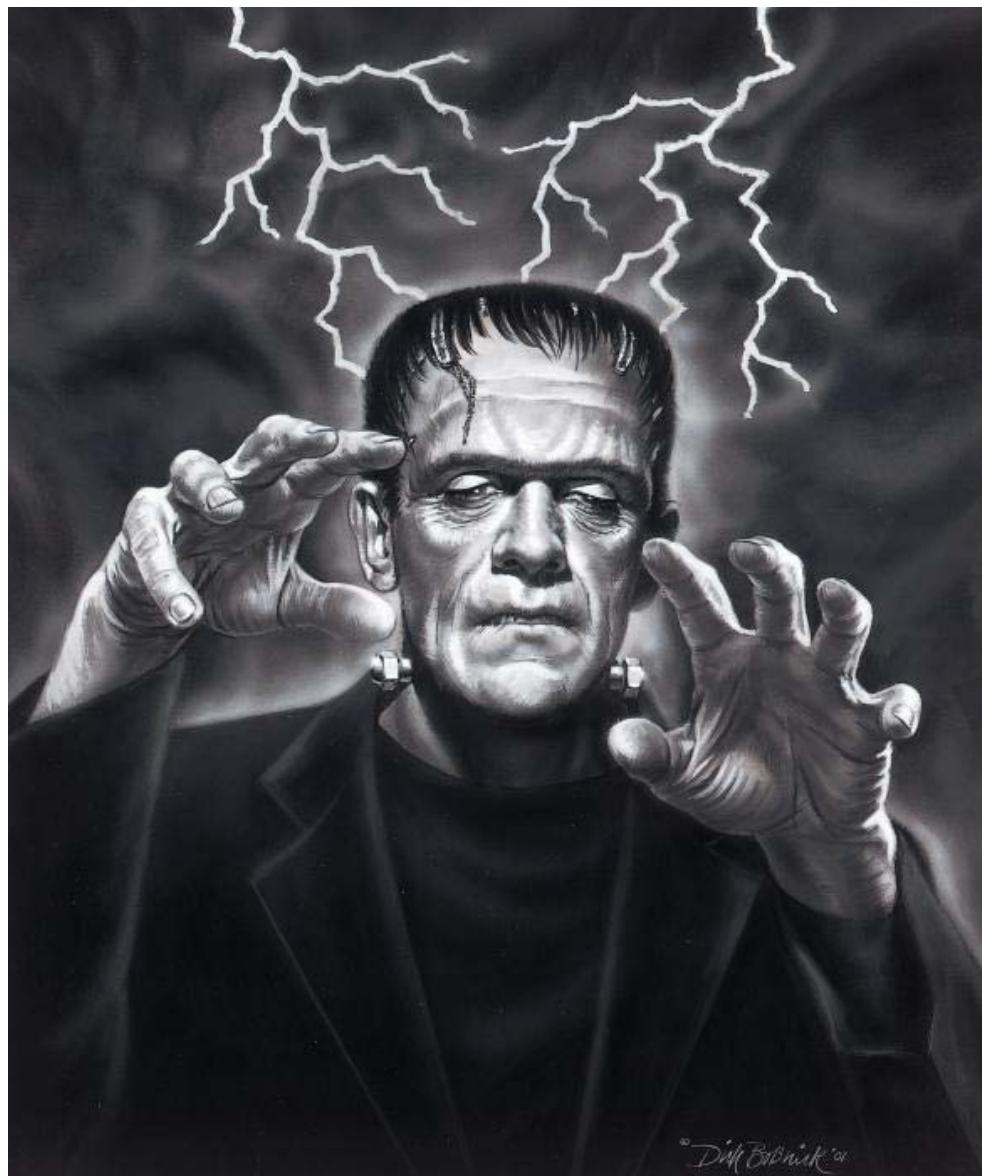
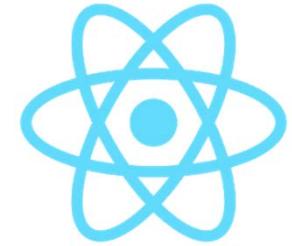


Mobile development

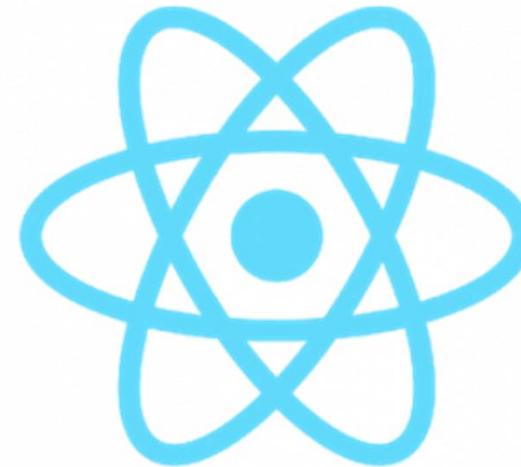


...

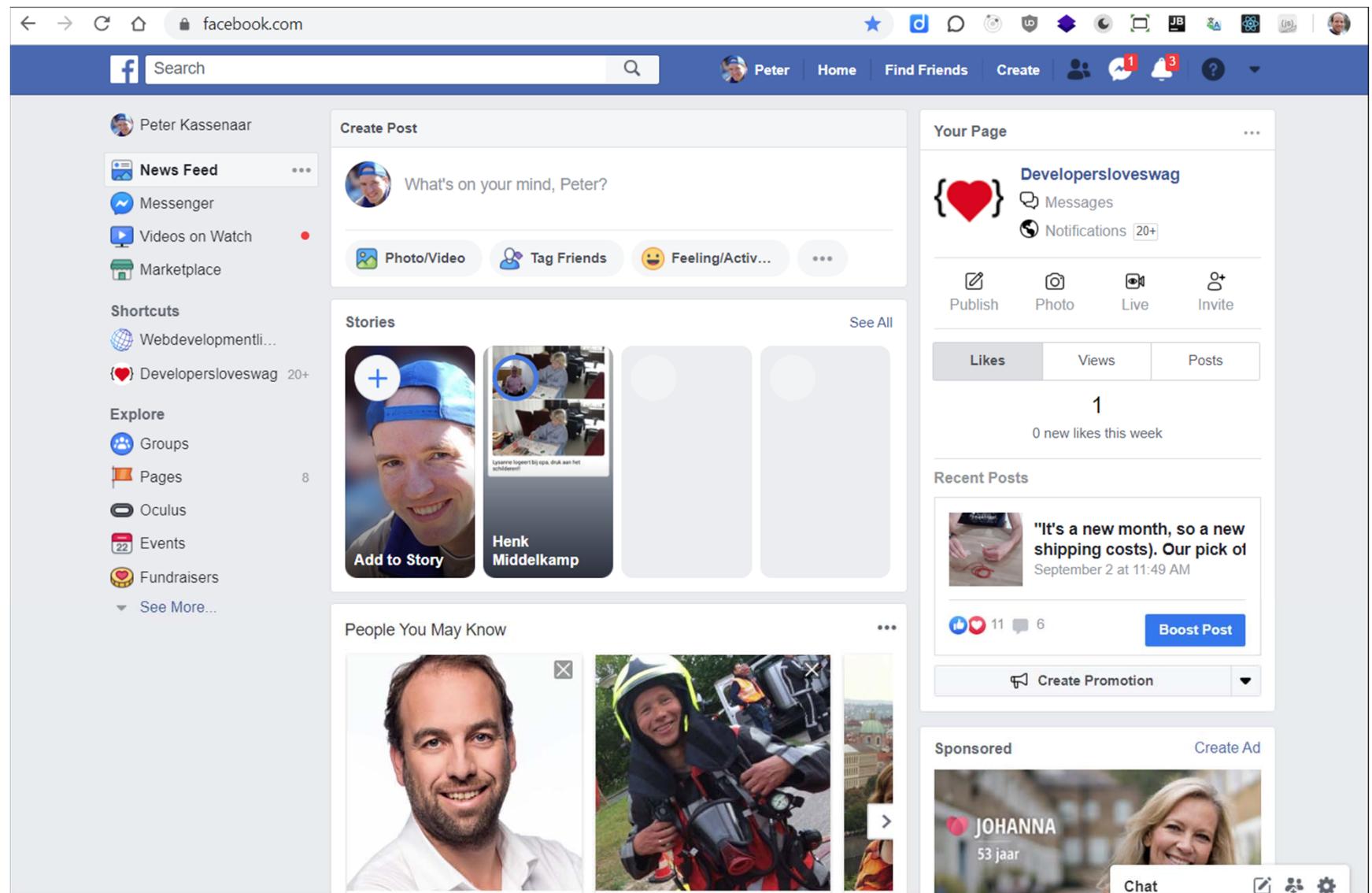
# “The Frankenstein Framework”



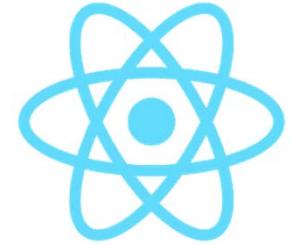
# Front-end Frameworks – the big four



# React – Created by Jordan Walke (Facebook)



# Brief history of React



2011 – Created at Facebook

2012 – Used on Instagram

2013 – Open source (May 29, 2013)

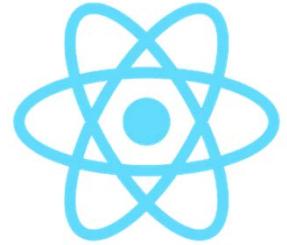
2014 – growth in popularity outside Facebook

2015 – React Native

2016 – React 15 (update to SemVer, from 0.14)

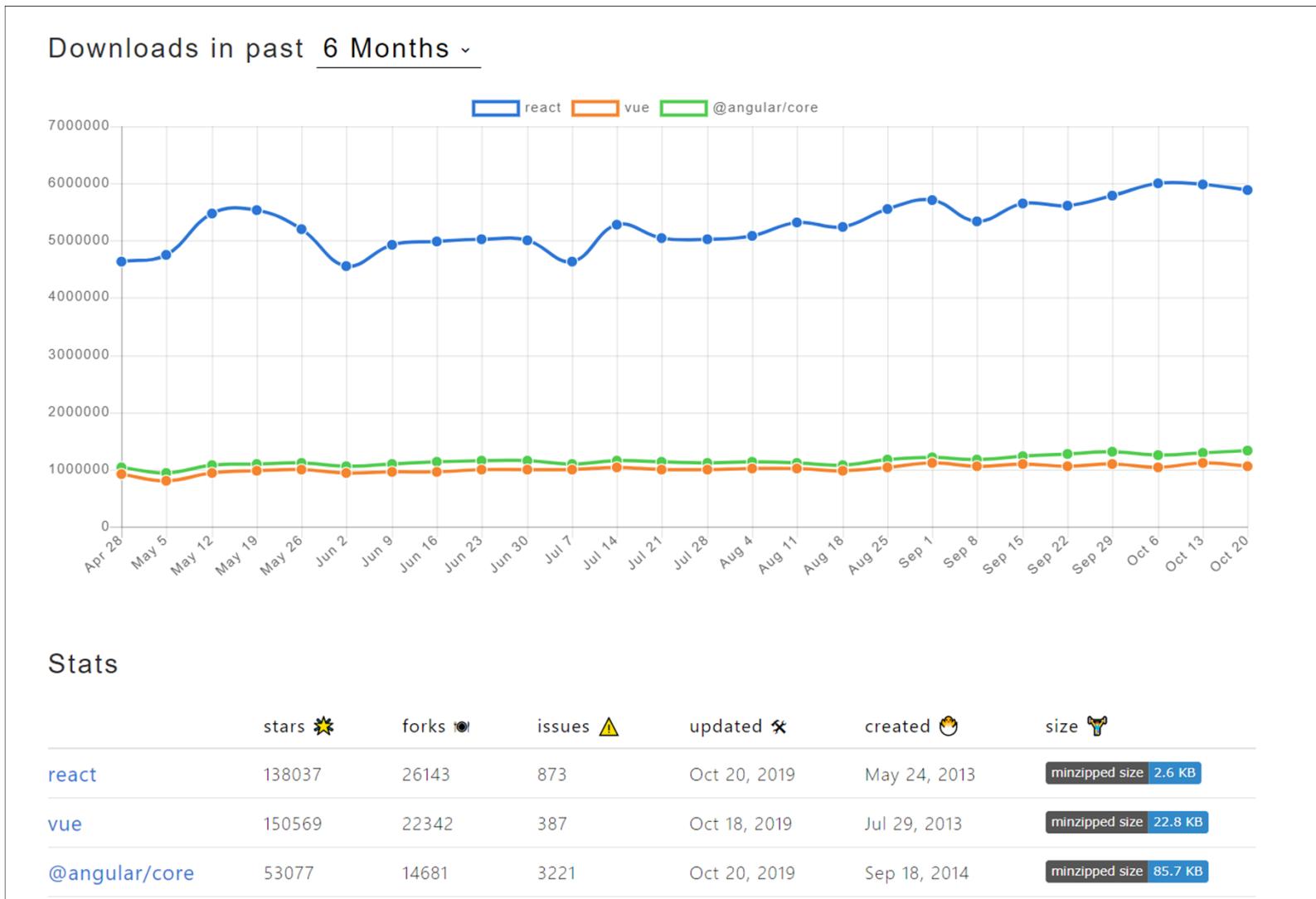
Today – *Facebook uses 30k+ components, employs full-time dev staff, used by many companies*

# Why React?



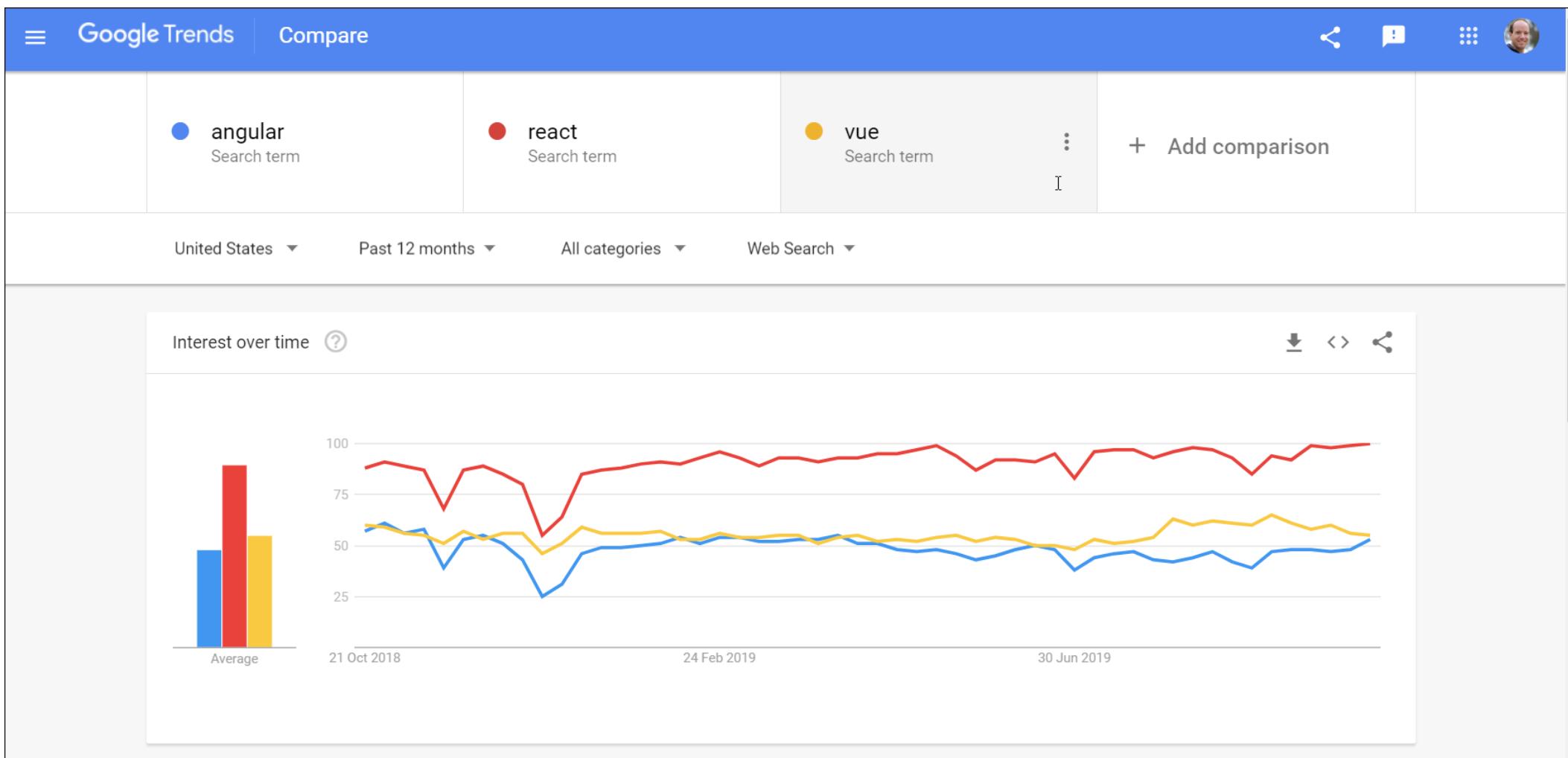
- Flexibility
- Developer experience
- Corporate investment
- Large community (Github, Stack Overflow)
- Performance
- Testability

# Npm trends comparison



<https://www.npmtrends.com/@angular/core-vs-react-vs-vue>

# Google Trends



<https://trends.google.com/trends/explore?geo=US&q=angular,react,vue>

# Giststar rankings

Gitstar Ranking    Users    Organizations    Repositories    GitHub username    Search    Sign in with GitHub

## Repositories Ranking

1 2 3 ... >

(Δ) 1. freeCodeCamp/freeCodeCamp	★ 298396
▼ 2. vuejs/vue	★ 129161
(B) 3. twbs/bootstrap	★ 128833
(F) 4. facebook/react	★ 124937
↑ 5. tensorflow/tensorflow	★ 123450
Ebook Foundation 6. EbookFoundation/free-progra...	★ 118765
OWL 7. sindresorhus/awesome	★ 99459
DB 8. d3/d3	★ 83265
▲ 51. zeit/next.js	★ 35904
⌚ 52. moment/moment	★ 35598
⬢ 53. nodejs/node-v0.x-archive	★ 35577
Ionic Team 54. ionic-team/ionic	★ 35555
Typicode 55. typicode/json-server	★ 35389
mrdoob 56. mrdoob/three.js	★ 35013
Rust Lang 57. rust-lang/rust	★ 34749
shadowsocks 58. shadowsocks/shadowsocks-win...	★ 34747

<https://gitstar-ranking.com/repositories>

CodeinWP content is free. When you purchase through referral links on our site, we earn a commission. [Learn more](#)

## Angular vs React vs Vue: Which Framework to Choose in 2020

by  Shaumik Daityari / updated: august 9, 2020 / [web & app frameworks](#)

This post is a comprehensive guide on which is perhaps the right solution for you: **Angular vs React vs Vue**.

Just a couple of years ago, developers were mainly debating on whether they should be using Angular vs React for their projects. But over the course of the last couple of years, we saw a growth of interest in a third player called Vue.js.

If you are a [developer starting out](#) on a project and cannot decide on which JavaScript framework to use, this guide should help you make a decision.

We cover various aspects of Angular, Vue, and React to see how they suit your needs. This post is not just a guide on Angular vs React vs Vue but aims to provide a structure to help judge front-end JavaScript frameworks in general. In case a new framework

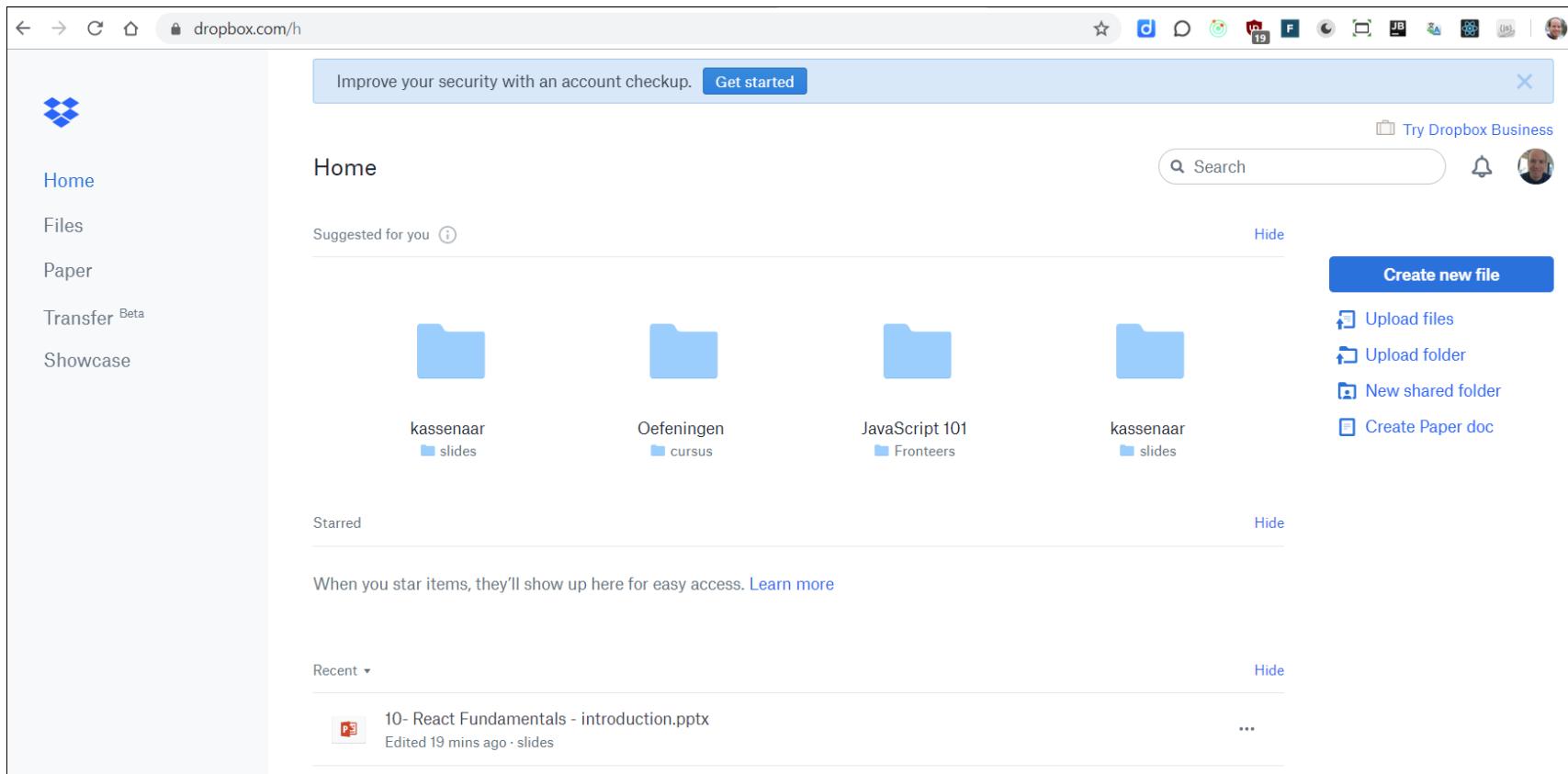


Which #javascript framework to choose in 2019: @angular vs @vuejs vs @reactjs

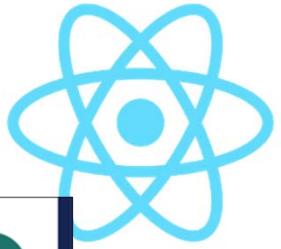
[CLICK TO TWEET](#) 

<https://www.codeinwp.com/blog/angular-vs-vue-vs-react/>

*“React is used by many Fortune 500 companies, including Airbnb, BBC, Instagram and Dropbox”*



# Google for 'built with react'



The screenshot shows a blog post from AnyforSoft Amplify Digital. At the top left is a "Menu" link. In the top center is the AnyforSoft logo with the text "AnyforSoft Amplify Digital". At the top right is a green "Hire us" button. The main title of the post is "10 Famous Websites Built with React JS". Below the title is the date "27 June 2019 |". The background of the post features a stylized illustration of a computer monitor displaying a React logo, set against a backdrop of clouds, trees, and a bridge.

The React JS library has grown in popularity over the years. Based on MVC architecture, it provides developers with the ability to create dynamic web page templates (e.g. pages with an extensive amount of interactive elements on them). That's one of the major reasons that so many popular web resources are based on React JS. Let's take a more detailed look at the advantages of employing this prominent library and a list of top 10 websites that use React JS, so you can view specific examples.

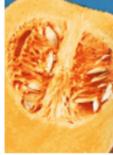
<https://anyforsoft.com/blog/10-famous-websites-built-react-js>

NL:

Inloggen ▾ Producten Bonus Allerhande Box Recepten Winkels Acties Meer ▾ Zoeken naar.. Online bestellen 0

## Deze herfst vallen ook de aanbiedingen van de boom.

[Naar Bonusaanbiedingen >](#)



**BONUS**

**3.49**  
1,35 kg

AH Herfst risotto verspakket

**30% KORTING**  
**1.81**  
per stuk

AH Biologisch Pompoen

**BONUS**  
**3.49**  
per stuk

AH Spruitjes pastinaak stampot verspakket

**2 VOOR 2.00**  
**1.19**  
150 g

AH Kleintje boerenkool

### Vanaf maandag in de Bonus



**3 stapelen voor 6.75**  
**2.99**  
300 g

M&M's Pinda

**stapelen tot 60%**  
**7.79**  
20 wasbeurten

Persil Wasmiddel color gel

**2 stapelen voor 2.00**  
**1.69**  
1 l

Appelsientje Sinaasappel

**25% KORTING**  
**2.96**  
225 g

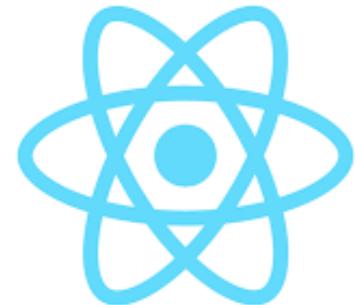
Old Amsterdam 8 plakken 48+

**1 + 1 GRATIS**  
**17.79**  
120 stuks

Davitamon Junior multivitamines ...

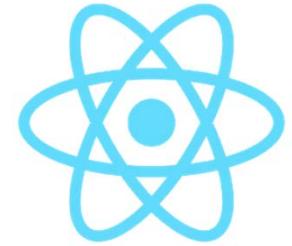
[Naar alle Bonusaanbiedingen >](#)

# Why \*not\* React?



- Framework (Angular, Vue)
- Concise (less code)
- Template centered
- Separate files
- Non-standard (html, elements, shadow DOM)
- Community backing (except Angular)
- Library
- Explicit (more coding)
- JavaScript Centered
- Single File
- Non standard (JSX, shadow DOM)
- Corporate backing

# More concerns...



*"But JSX is so different from HTML!"*

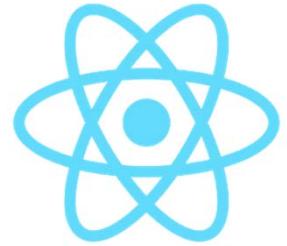
- True, but that's the choice React made
- Point of departure is **JavaScript**. Not the HTML-template

```
<h2 color="red">This is a heading</h2>
```



```
React.createElement('h2', { color: 'red' }, 'This is a heading');
```

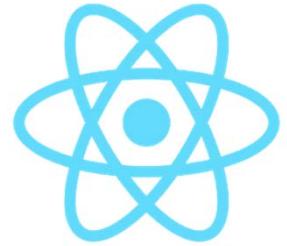
# JSX differences



However - JSX looks a lot like HTML. There are only a few differences:

HTML	JSX
for (for labels)	htmlFor
class	className
style="color: red"	<style={{color: 'red'}} >
<!-- comment -->	{ /* Comment */ }
Events: click, blur, focus etc	Prefix with on+capital: onClick, onBlur, onFocus, etc

# Still worried?



- HTML to JSX compiler
- <https://magic.reactjs.net/htmltojsx.htm>

## HTML to JSX Compiler

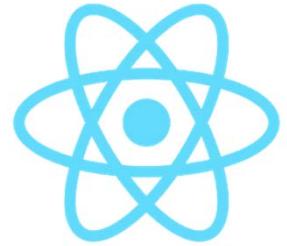
Create class: Class name: NewComponent

Live HTML Editor

```
<!-- Hello world -->
<div class="awesome" style="border: 1px solid red">
  <label for="name">Enter your name: </label>
  <input type="text" id="name" />
</div>
<p>Enter your HTML here</p>
```

```
var NewComponent = React.createClass({
  render: function() {
    return (
      <div>
        {/* Hello world */}
        <div className="awesome" style={{border: '1px solid red'}}>
          <label htmlFor="name">Enter your name: </label>
          <input type="text" id="name" />
        </div>
        <p>Enter your HTML here</p>
      </div>
    );
  }
});
```

# Other concerns...

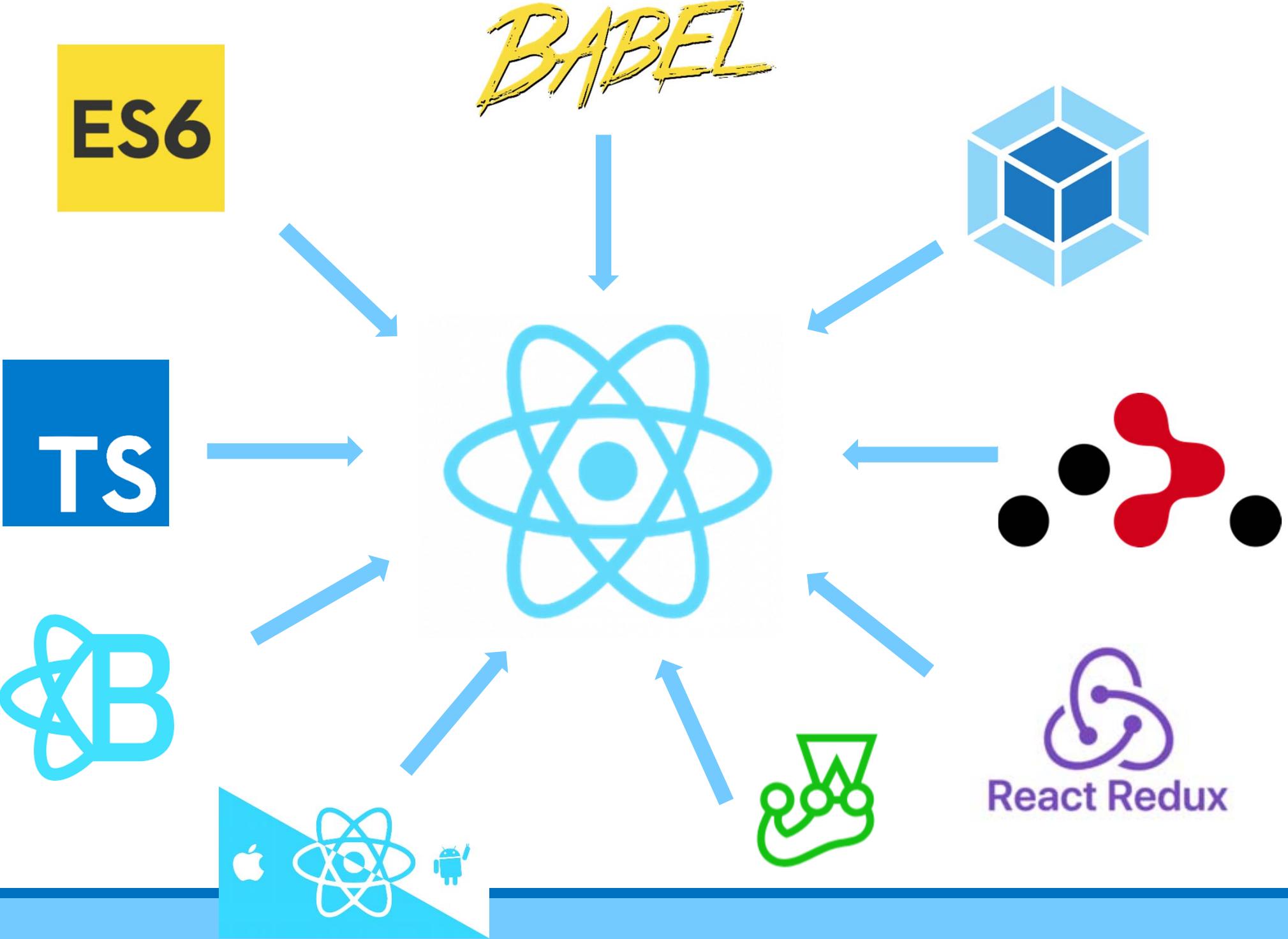


- “React requires a build step”
  - True, but this is also true for all other modern frameworks.
- “There might be version conflicts”
  - Yes, you need to address that and update the code. Just as in other frameworks.
  - Use <https://github.com/reactjs/react-codemod> to update code
  - Dependencies (router, redux) should be updated separately
- “But there is so much old and outdated stuff online”
  - True. As with other frameworks. Always check date and validity.



# The React ecosystem

React is not isolated/not living on an island



# Additional services, options, etc.

The image is a collage of three screenshots of developer tools and services:

- Amplify Framework:** A screenshot of the Amplify Framework website. It features a large orange header with the text "The foundation for your cloud-powered mobile & web apps". Below the header are icons for Apple, Android, and React. A "GET STARTED" button is visible. The top navigation bar includes "Get Started", "Toolchain", and "St".
- Cloudinary:** A screenshot of the Cloudinary website. It has a dark header with "Cloudinary" and "SOLUTIONS", "PRICING", and "CLOUDINARY" buttons. The main content area shows a blurred image of a person working on a laptop.
- React Training:** A screenshot of the React Training website. It features a logo with a red atom icon and the text "REACT TRAINING". Below the logo is the heading "Expert React Training" and the subtext "From the creators of React Router and Reach UI". The top navigation bar includes "REACT TRAINING", "ATTEND A WORKSHOP", "CORPORATE TRAINING", and "BLOG". A "EXPLORE EXAMPLES" button is visible on the right.

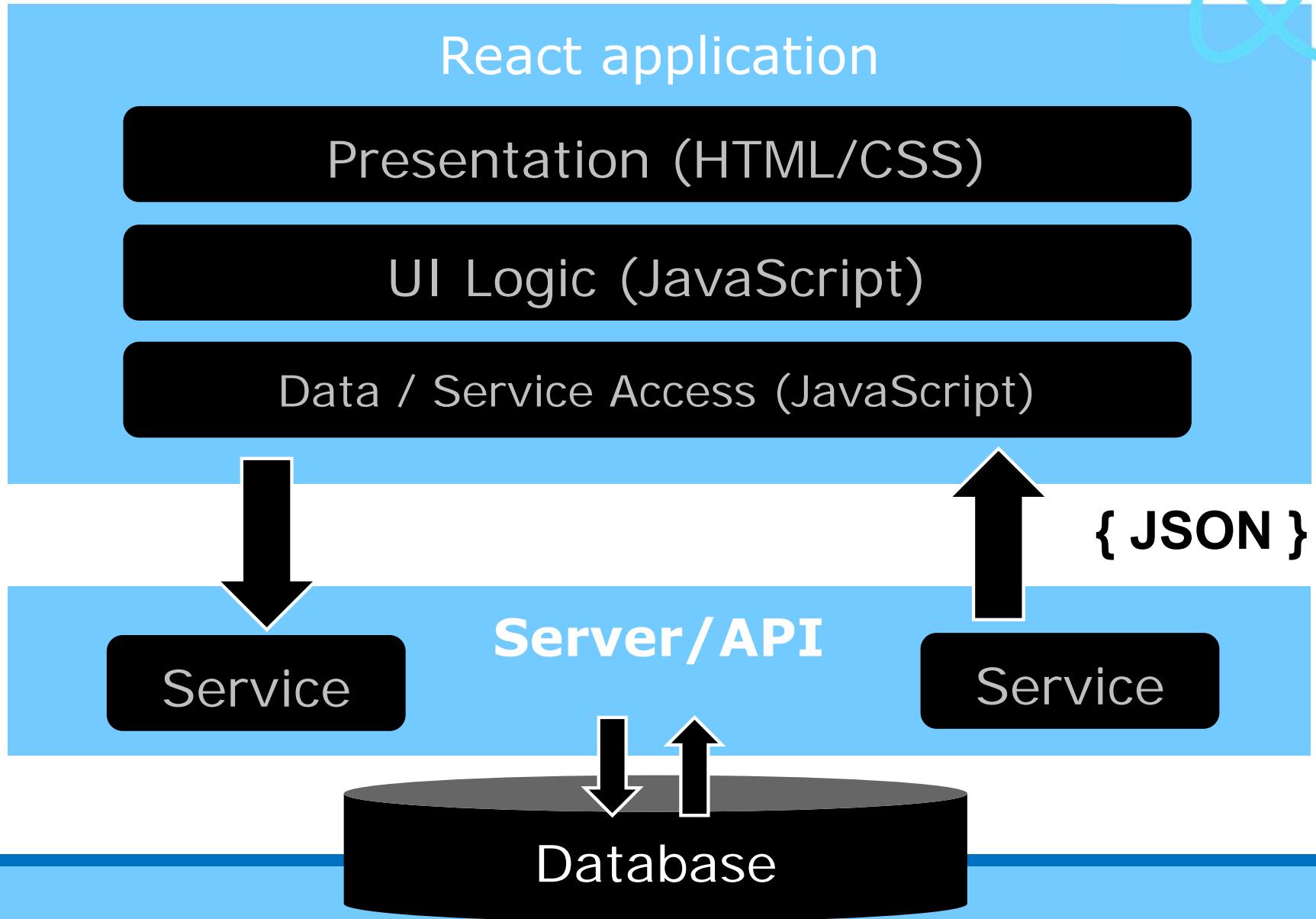
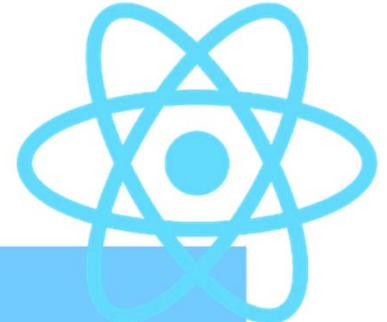
And more....



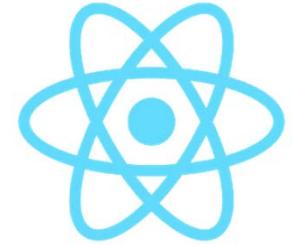
# React's basic concepts

Getting a conceptual understanding of every React app

# Single Page Application



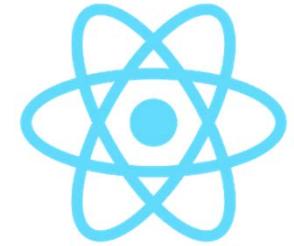
# 3 Basic principles in every React App



## 1. Components

- Like functions: class based and function based components
- Input: props and state
- Output: UI (via the `render()` function)
- Reusable and composable
- Mentioned via their name, like `<myComponent />`
- Components can have private state

# 3 Basic principles in every React App



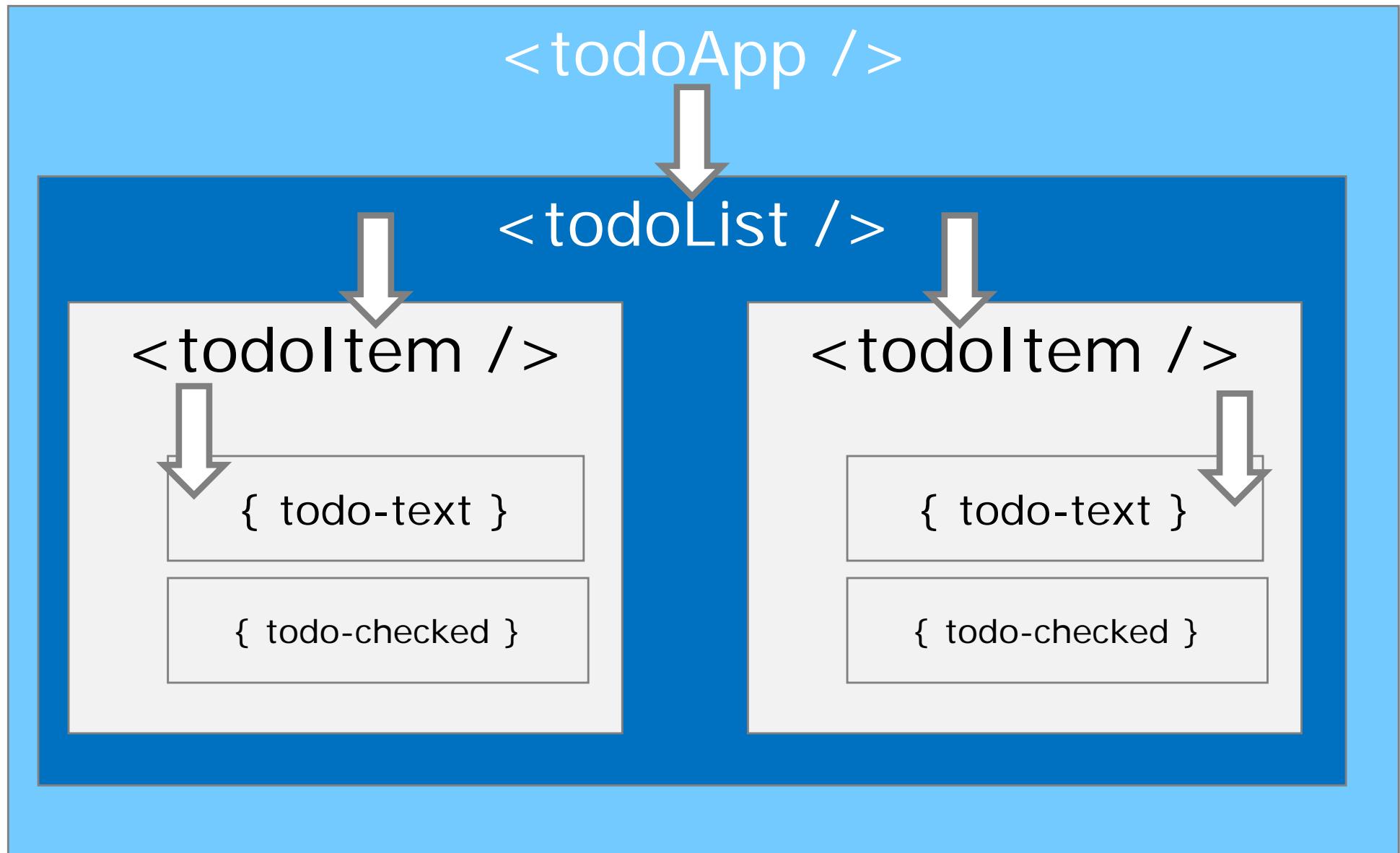
## 2. Reactive updates

- React will react to the changes it makes or receives
- Update the browser, based on those changes

## 3. Virtual views in memory

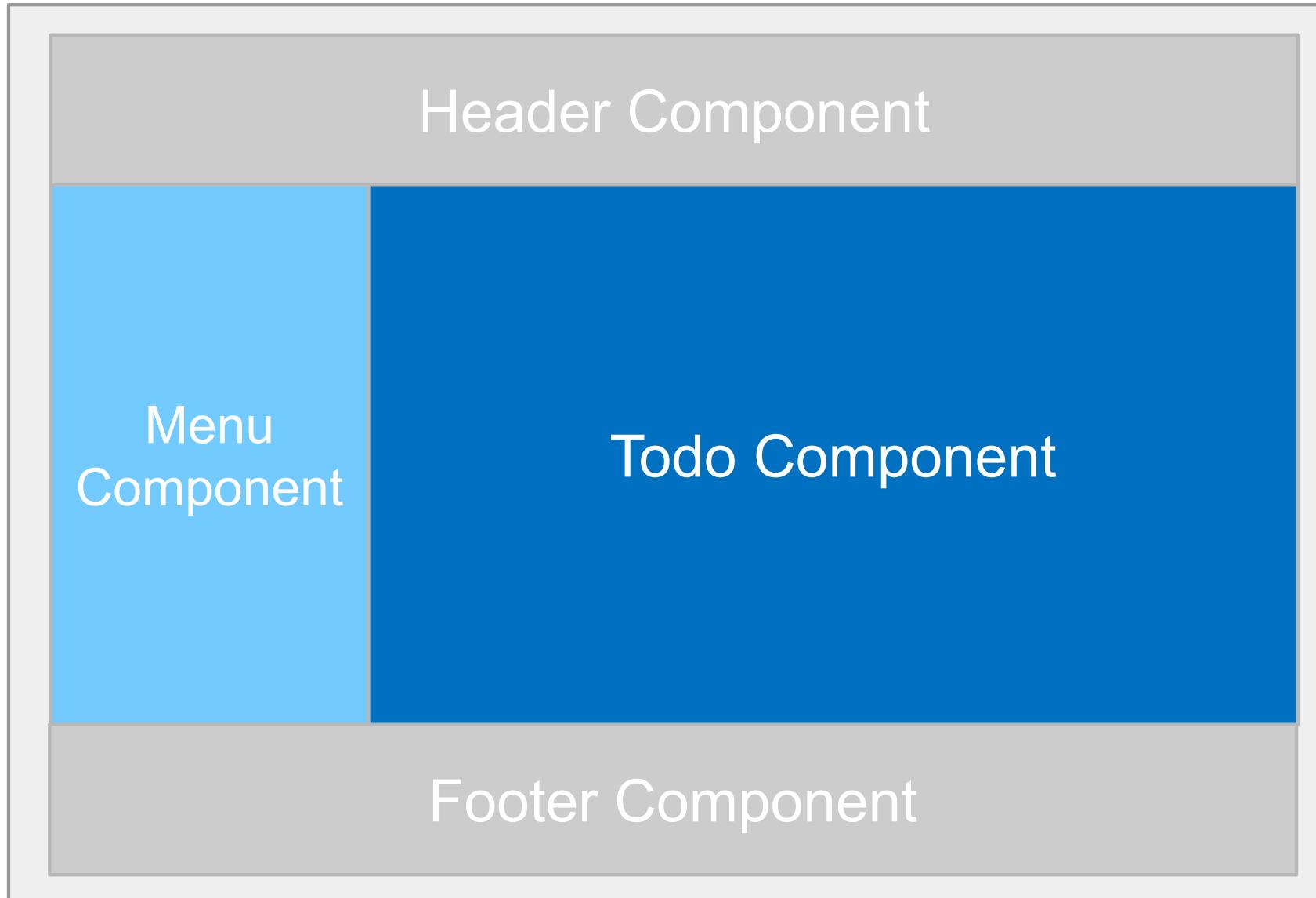
- Don't write HTML (for instance to handle data)
- Use JavaScript (or JSX) to generate HTML
- No HTML template language
- Virtual DOM or Tree reconciliation
- Build the DOM in-memory before adding it to the UI

# React – app- & component structure



*“A React app is a tree of components”*

# Components – visual representation



# React components

## Function based components

Simple – just write functions that return the UI for the component

## Class based components

A bit more complex – but also more powerful (? - opinion)

*You will use – and need to learn – both*

```
const Counter = props => {
  const [counter, setCounter] = useState(0)
  return (
    <div>
      <h1>A counter component</h1>
    </div>
  )
};
```

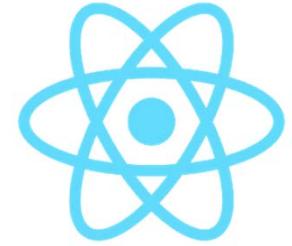
Function component

```
class Counter extends React.Component {
  state = {
    counter: 0
  };

  render() {
    return (
      <div>
        <h1>A counter component</h1>
      </div>
    )
  }
};
```

Class component

# Compiling a class component



JSX is compiled to JavaScript

(here: the class component):

```
class Counter extends React.Component {  
  state = {  
    counter: 0  
  };  
  
  render() {  
    return (  
      React.createElement('div', null,  
        React.createElement('h1', null, 'A counter component')  
      )  
    )  
  }  
};
```

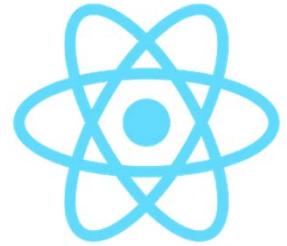
# Questions?



# Let's create an app

Install tooling and create your first app

# Install necessary Tooling



First – install Git, if you don't have it yet

The screenshot shows the official Git website at <https://git-scm.com/downloads>. The main navigation bar includes links for About, Documentation, Downloads (selected), GUI Clients, Logos, and Community. A sidebar mentions the "Pro Git book" by Scott Chacon and Ben Straub. The central "Downloads" section features sections for Mac OS X, Windows, and Linux/Unix. A large image of a Mac monitor displays the latest source release version 2.19.2. Below the monitor, a terminal window shows the command "git --version" being run, with the output "git version 2.17.2 (Apple Git-113)" highlighted with a red oval. The user's name "PeterKassenaar" is visible in the terminal window.

git --fast-version-control

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire [Pro Git book](#) written by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).

## Downloads

Mac OS X Windows

Linux/Unix

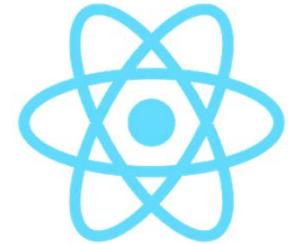
Older releases are available and the [Git source repository](#) is on GitHub.

### GUI Clients

Git comes with built-in clients (gitk, git gui, gitk), but there are several other clients for users looking for a platform-independent experience.

Last login: Mon Nov 26 13:28:59 on ttys011  
MacBook-Pro:~ PeterKassenaar\$ git --version  
git version 2.17.2 (Apple Git-113)  
MacBook-Pro:~ PeterKassenaar\$

# NodeJS



The Node.js logo, featuring the word "node" in a lowercase sans-serif font above a hexagonal icon containing the letters "js".

HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | NEWS | FOUNDATION

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

Download for macOS (x64)

**10.13.0 LTS**  
Recommended For Most Users

**11.2.0 Current**  
Latest Features

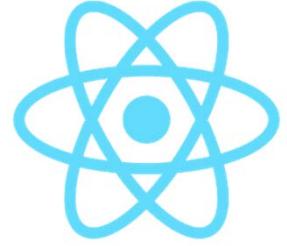
[Other Downloads](#) | [Changelog](#) | [API Docs](#)    [Other Downloads](#) | [Changelog](#) | [API Docs](#)

Or have a look at the [Long Term Support \(LTS\) schedule](#).

Sign up for [Node.js Everywhere](#), the official Node.js Monthly Newsletter.

<https://nodejs.org/en/>

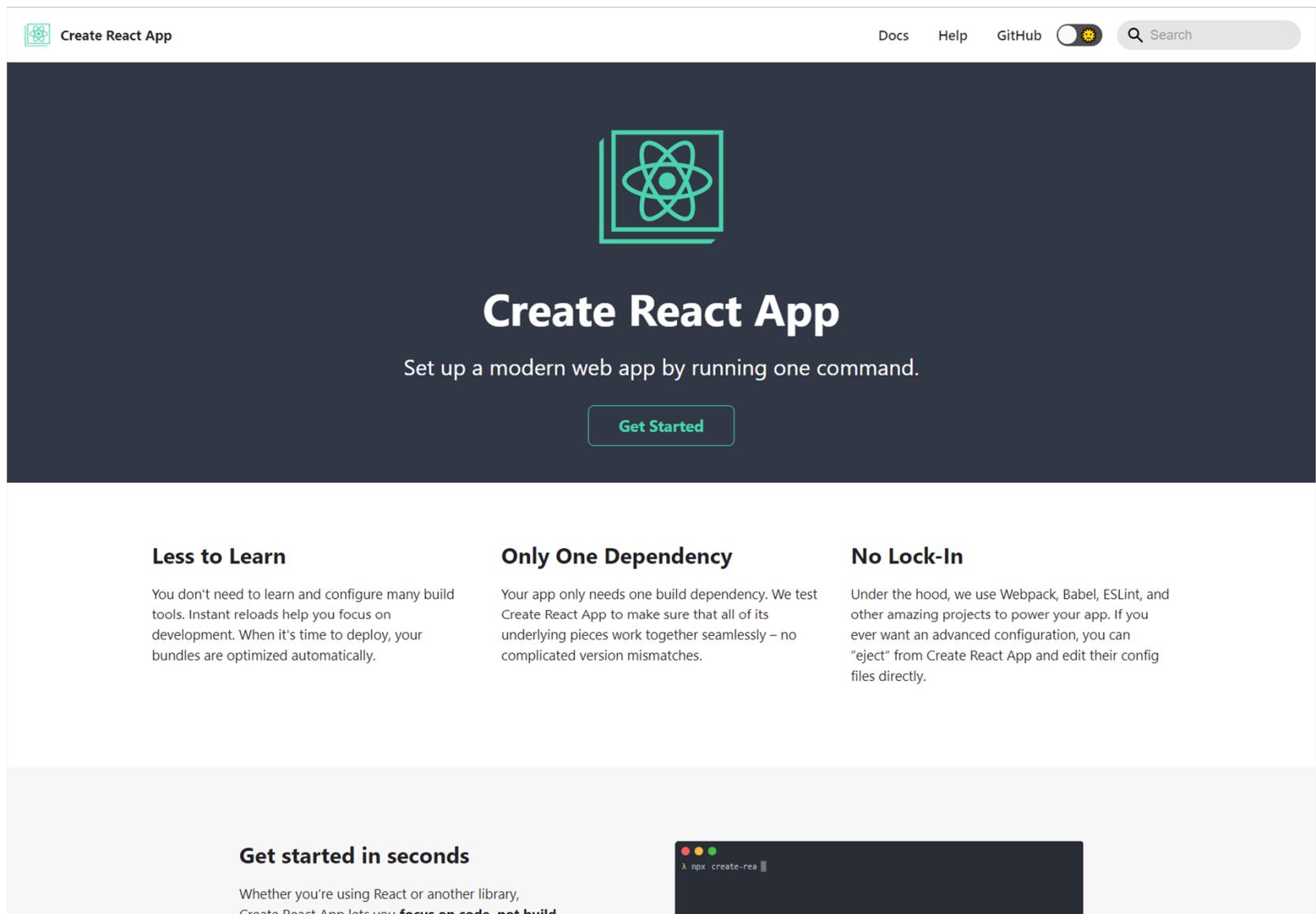
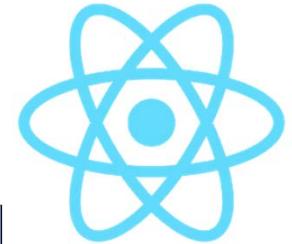
# create-react-app



- Create React App (or 'CRA' for short)
  - Scaffold new projects
  - Set up JSX compilation by Babel
  - Set up Webpack bundling
  - Set up Linting
  - ...
- All this is done by CRA
- <https://create-react-app.dev/>

```
npx create-react-app my-app
```

# create-react-app.dev



The screenshot shows the homepage of [create-react-app.dev](https://create-react-app.dev). At the top, there's a navigation bar with links for "Docs", "Help", "GitHub", and a search bar. Below the header is a large central area featuring the React logo icon, followed by the text "Create React App" and a subtext "Set up a modern web app by running one command." A prominent "Get Started" button is centered below this text. The main content area is divided into three columns: "Less to Learn", "Only One Dependency", and "No Lock-In". Each column contains a brief description of the benefit. At the bottom left, there's a section titled "Get started in seconds" with a note about the focus on code over build tools, and a terminal window showing the command "npx create-react-app".

Create React App

Docs Help GitHub Search

Get Started

**Less to Learn**

You don't need to learn and configure many build tools. Instant reloads help you focus on development. When it's time to deploy, your bundles are optimized automatically.

**Only One Dependency**

Your app only needs one build dependency. We test Create React App to make sure that all of its underlying pieces work together seamlessly – no complicated version mismatches.

**No Lock-In**

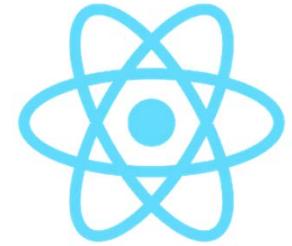
Under the hood, we use Webpack, Babel, ESLint, and other amazing projects to power your app. If you ever want an advanced configuration, you can "eject" from Create React App and edit their config files directly.

**Get started in seconds**

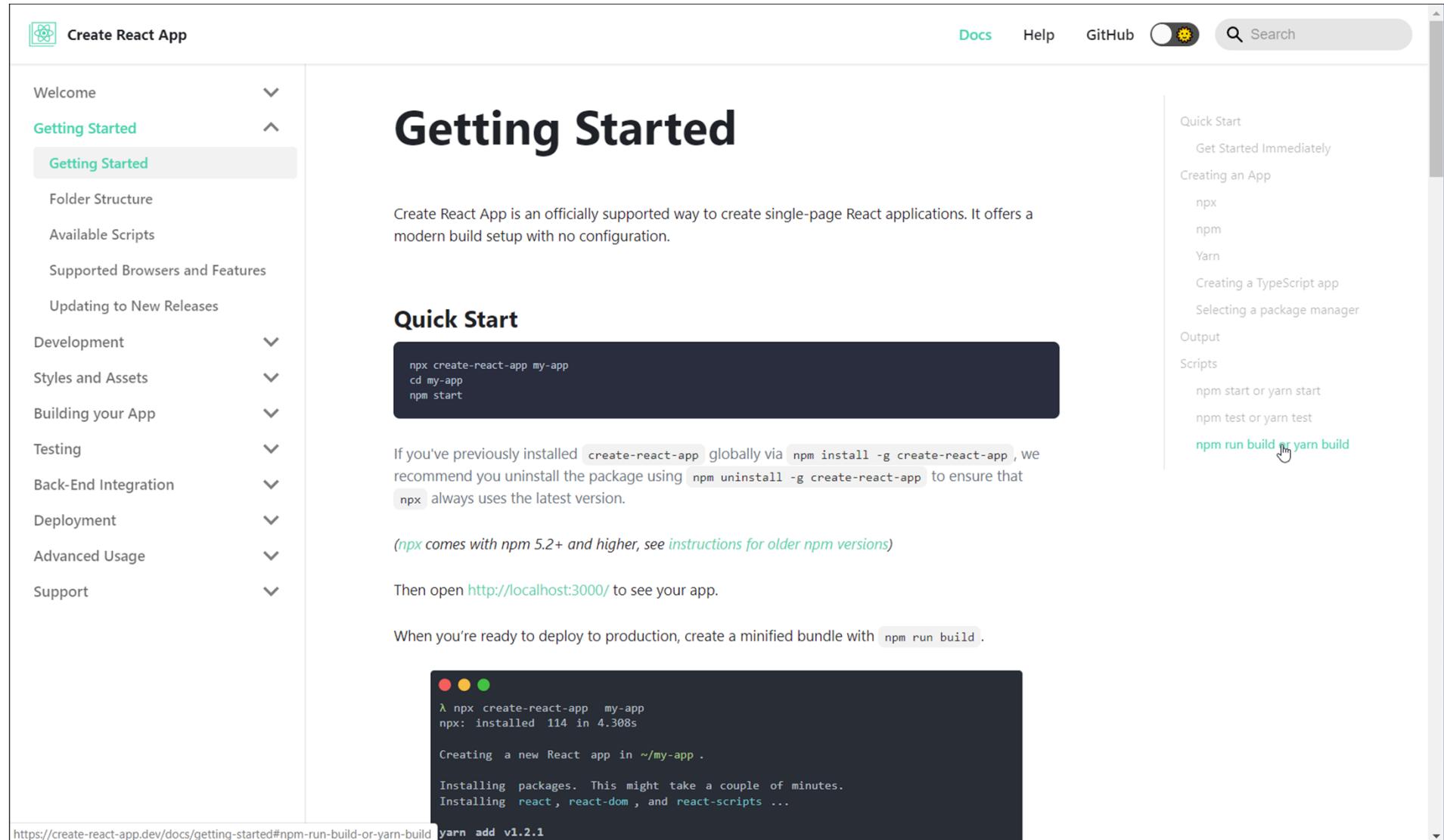
Whether you're using React or another library, [Create React App lets you focus on code, not build](#)

```
npx create-react-app
```

# Always check the documentation



https://create-react-app.dev/docs/getting-started#npm-run-build-or-yarn-build `yarn add v1.2.1`



The screenshot shows the 'Getting Started' section of the Create React App documentation. On the left, there's a sidebar with a tree-like navigation menu. The 'Getting Started' section is expanded, showing 'Getting Started' (selected), 'Folder Structure', 'Available Scripts', 'Supported Browsers and Features', 'Updating to New Releases', 'Development', 'Styles and Assets', 'Building your App', 'Testing', 'Back-End Integration', 'Deployment', 'Advanced Usage', and 'Support'. The main content area has a large heading 'Getting Started' and a paragraph about the benefits of using Create React App. Below that is a 'Quick Start' section with a code block containing the command `npx create-react-app my-app`. Further down, it says if you've installed `create-react-app` globally, you should uninstall it and use `npx` instead. It also notes that `npx` requires npm 5.2+ and provides a link for older versions. It then instructs to open `http://localhost:3000/` and shows a terminal window running the command. On the right side, there's a sidebar with links like 'Quick Start', 'Get Started Immediately', 'Creating an App', 'npx', 'npm', 'Yarn', 'Creating a TypeScript app', 'Selecting a package manager', 'Output', 'Scripts', 'npm start or yarn start', 'npm test or yarn test', and 'npm run build' (which has a hand cursor icon over it).

Create React App is an officially supported way to create single-page React applications. It offers a modern build setup with no configuration.

## Quick Start

```
npx create-react-app my-app  
cd my-app  
npm start
```

If you've previously installed `create-react-app` globally via `npm install -g create-react-app`, we recommend you uninstall the package using `npm uninstall -g create-react-app` to ensure that `npx` always uses the latest version.

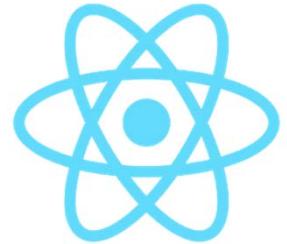
(`npx` comes with npm 5.2+ and higher, see [instructions for older npm versions](#))

Then open `http://localhost:3000/` to see your app.

When you're ready to deploy to production, create a minified bundle with `npm run build`.

```
λ npx create-react-app my-app  
npx: installed 114 in 4.308s  
  
Creating a new React app in ~/my-app .  
  
Installing packages. This might take a couple of minutes.  
Installing react, react-dom, and react-scripts ...
```

# CRA



- Use `npx` to run it. You *can* install CRA locally, but this is not recommended
- You don't need to set up webpack and babel yourself

```
npm
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. Alle rechten voorbehouden.

C:\Users\Gebruiker>cd Desktop

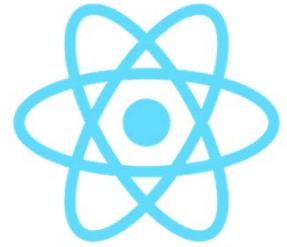
C:\Users\Gebruiker\Desktop>npx create-react-app my-app
npx: installed 91 in 6.344s

Creating a new React app in C:\Users\Gebruiker\Desktop\my-app.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts...

[████.....] | fetchMetadata: sill resolveWithNewModule color-convert@1.9.3 checking installable status
```

# Creating a default application



```
Opdrachtprompt
Initialized a git repository.

Success! Created my-app at C:\Users\Gebruiker\Desktop\my-app
Inside that directory, you can run several commands:

npm start
  Starts the development server.

npm run build
  Bundles the app into static files for production.

npm test
  Starts the test runner.

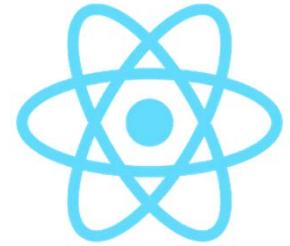
npm run eject
  Removes this tool and copies build dependencies, configuration files
  and scripts into the app directory. If you do this, you can't go back!

We suggest that you begin by typing:

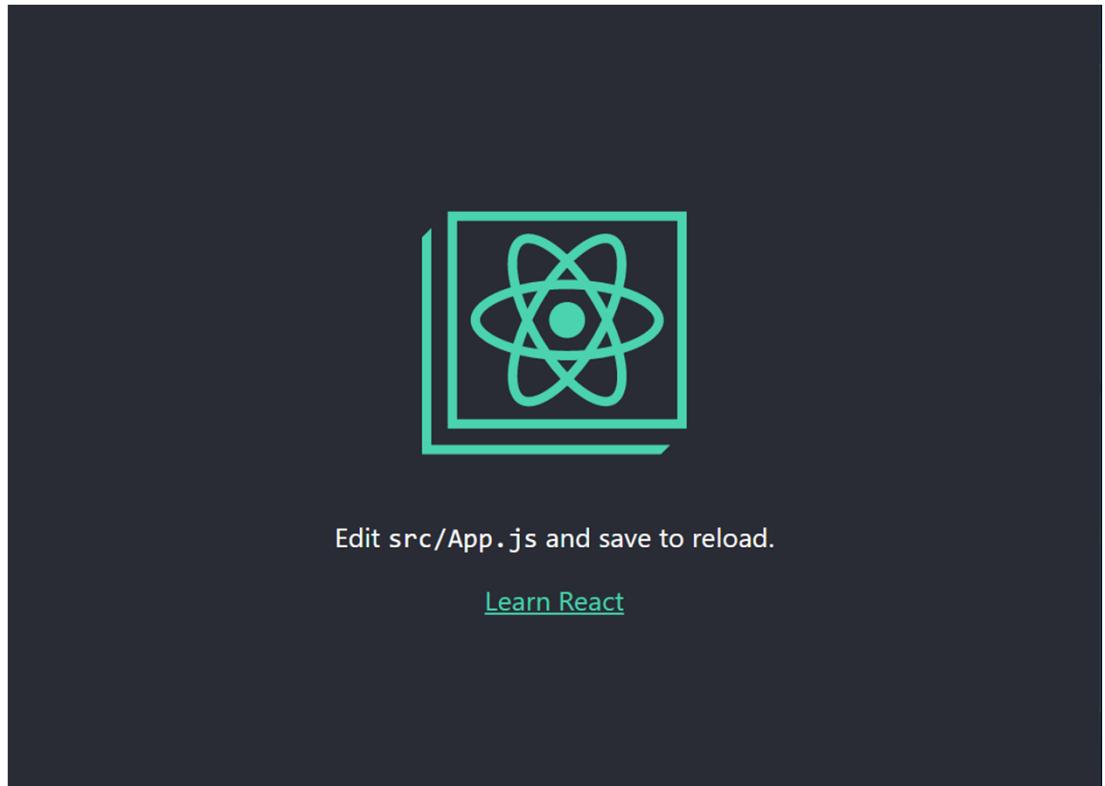
cd my-app
npm start

Happy hacking!
C:\Users\Gebruiker\Desktop>cd my-app
C:\Users\Gebruiker\Desktop\my-app>npm start
```

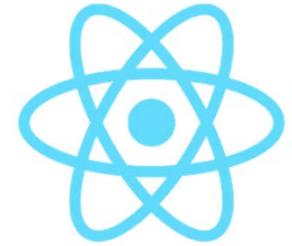
# Running the default project



- npm start
- <http://localhost:3000>



# File structure



The screenshot shows a code editor interface with a project structure on the left and the content of `App.js` on the right.

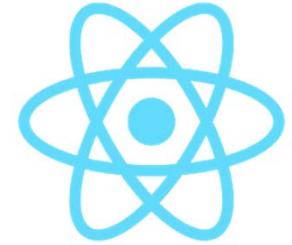
**Project Structure:**

- my-app
- src
  - App.css
  - App.js
  - App.test.js
  - index.css
  - index.js
  - logo.svg
  - serviceWorker.js
- .gitignore
- package.json
- package-lock.json
- README.md
- External Libraries
- Scratches and Consoles

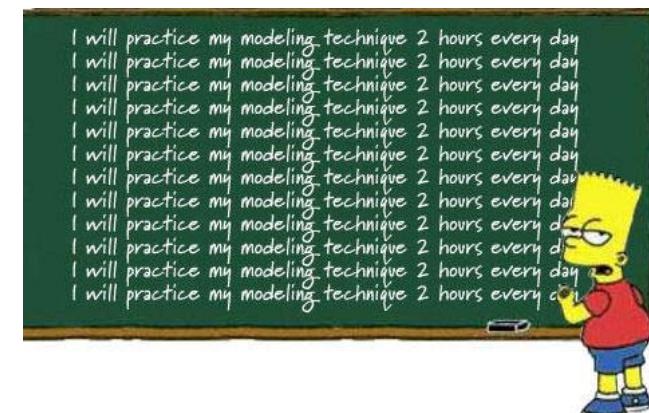
**Code Editor Content (`App.js`):**

```
1 import React from 'react';
2 import logo from './logo.svg';
3 import './App.css';
4
5 function App() {
6   return (
7     <div className="App">
8       <header className="App-header">
9         <img src={logo} className="App-logo" alt="logo" />
10        <p>
11          Edit <code>src/App.js</code> and save to reload.
12        </p>
13        <a
14          className="App-link"
15          href="https://reactjs.org"
16          target="_blank"
17          rel="noopener noreferrer"
18        >Get started</a>
19      </header>
20      <main className="App-main">
21        <h1>Hello world!</h1>
22        <p>This is a React component</p>
23      </main>
24    </div>
25  )
26}
27
28export default App;
```

# Workshop



- Create a default project `npx create-react-app my-app`
  - <https://create-react-app.dev/docs/getting-started> (for background info)
- Create a new, blank React project
- Run it
  - `npm start`
- Replace some content in `src/App.js` component with your own code.
- Code example: [..../100-helloworld](#)





# Let's look at the code

A look inside the key files in the generated code

# #1. Package.json



The screenshot shows a code editor window with the title "package.json". The file contains the configuration for a React application. The code is color-coded: green for strings and blue for boolean values. The structure includes dependencies on react, react-dom, and react-scripts, and scripts for start, build, test, and eject. It also specifies an eslintConfig and a browserslist for production.

```
1  "name": "hello-react",
2  "version": "0.1.0",
3  "private": true,
4  "dependencies": {
5    "react": "^16.11.0",
6    "react-dom": "^16.11.0",
7    "react-scripts": "3.2.0"
8  },
9  "scripts": {
10    "start": "react-scripts start",
11    "build": "react-scripts build",
12    "test": "react-scripts test",
13    "eject": "react-scripts eject"
14  },
15  "eslintConfig": {
16    "extends": "react-app"
17  },
18  "browserslist": {
19    "production": [
20      ">0.2%"
21    ]
22  }
23 }
```

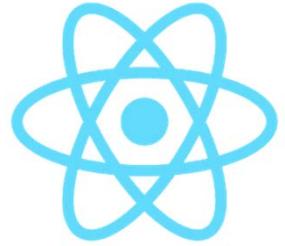
## #2 ..src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import * as serviceWorker from './serviceWorker';

ReactDOM.render(<App />, document.getElementById('root'));

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();
```

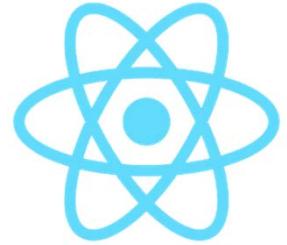
# index.js



- Is the root of your application
- Imports starting component
  - Defaults to: <App />
- Adds App to the root element in index.html
- Typically you *never touch* this file again

```
ReactDOM.render(  
  <App />,  
  document.getElementById('root')  
) ;
```

# #3. App.js



- <App /> is the main component of the app.
  - You typically replace the contents with your own code
  - Main Navigation, routing, other content, footer, etc.

```
import React from 'react';
import logo from './logo.svg';
import './App.css';
```

Import(s)

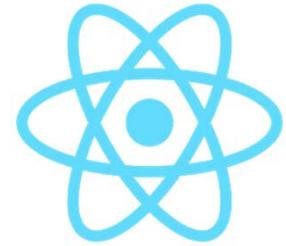
```
function App() {
  return (
    <div className="App">
      <header className="App-header">
        ...
      </header>
    </div>
  );
}
```

Component  
(function or  
class)

```
export default App;
```

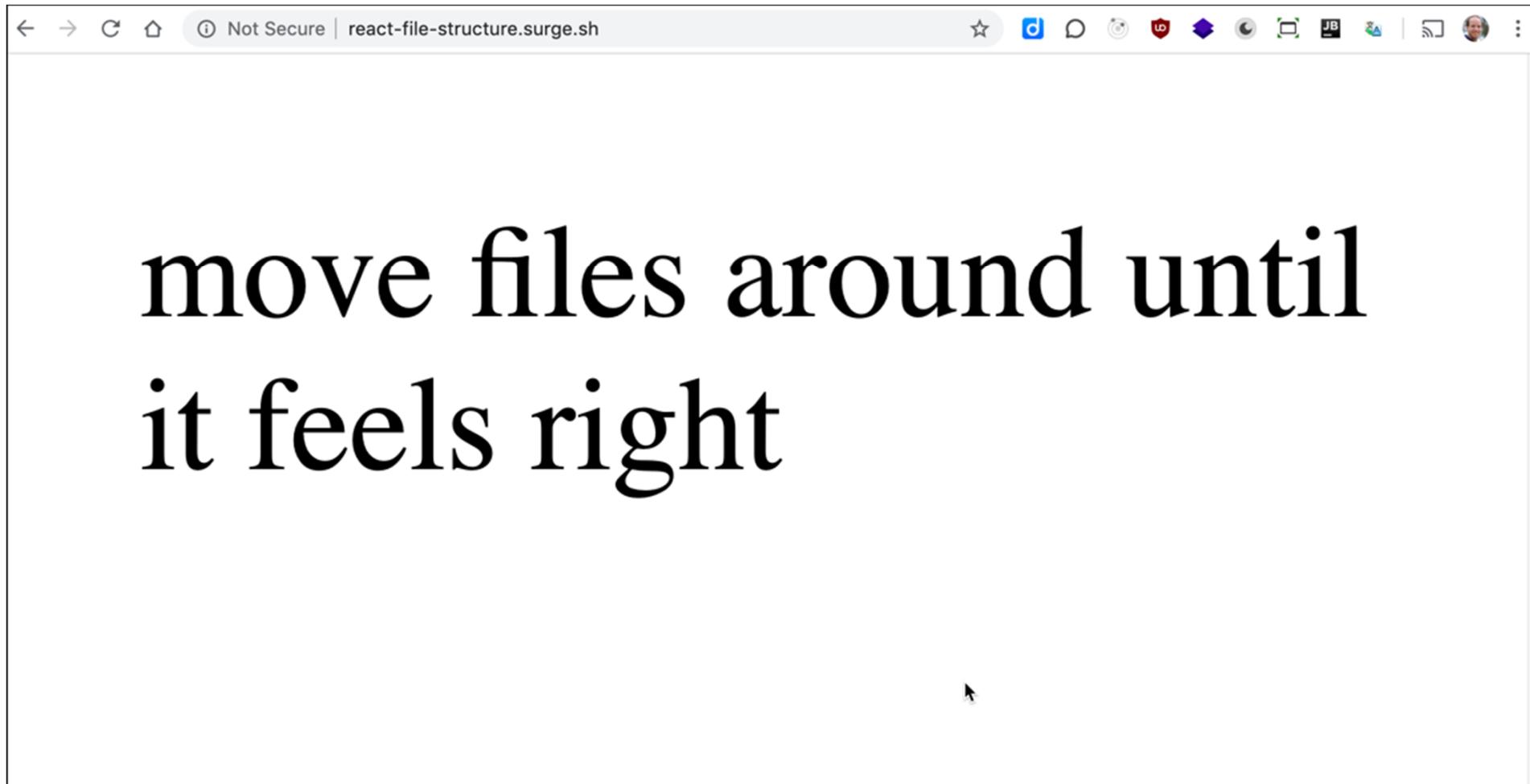
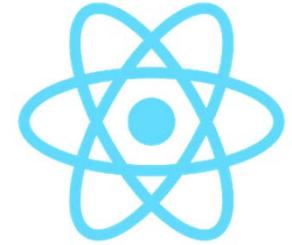
Export

## #4. Other files



- \* .css – various stylesheets.
- \* .test.js – various unit testing files.
- serviceWorker.js – to create a PWA. You may remove it.
- Folder /public – all files that you want to publish

# How to structure your React project?

A placeholder image showing a browser window with a blank white page. The browser interface includes a toolbar at the top with icons for back, forward, search, and other functions. The address bar shows the URL "react-file-structure.surge.sh" and indicates it is "Not Secure".

move files around until  
it feels right

Dan Abramov: <http://react-file-structure.surge.sh/>

# Some other opinions

 **React** Docs Tutorial Blog Community

## File Structure

**Is there a recommended way to structure React projects?**

React doesn't have opinions on how you put files into folders. That said there are a few common approaches popular in the ecosystem you may want to consider.

Grouping by features or routes

One common way to structure projects is locate CSS, JS, and tests together inside folders grouped by feature or route.

[reactjs.org/docs/faq-structure.html](https://reactjs.org/docs/faq-structure.html)



## How I structure a React project

 Maciek Chmura  May 28 Updated on Jun 05, 2019 · 2 min read

#javascript #react

There are many guides on how to structure web apps and React in particular.

- Move files into folders based on Smart/Dumb components.
- Organise them by the Header/Main/Footer
- Or throw everything to Components and combine them in Pages???

<https://dev.to/maciekchmura/how-i-structure-a-react-project-3c2i>

# Advanced tips on structuring larger applications

 sitepoint

Blog   Community   Jobs   Library   Login   [Join Premium](#)

## How to Organize a Large React Application and Make It Scale

By Jack Franklin   JavaScript   November 2, 2020

Share:   

This article is by guest author [Jack Franklin](#). SitePoint guest posts aim to bring you engaging content from prominent writers and speakers of the Web community.

In this article, I'll discuss the approach I take when building and structuring large React applications. One of the best features of React is how it gets out of your way and is anything but descriptive when it comes to file structure. Therefore, you'll find a lot of questions on Stack Overflow and similar sites asking how to structure applications. This is a very opinionated topic, and there's no one right way. In this article, I'll talk you through the decisions I make when building React applications: picking tools, structuring files, and breaking components up into smaller pieces.

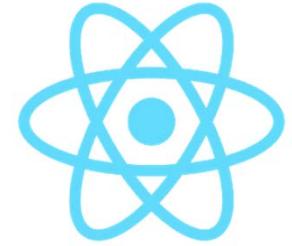
  
Free JavaScript Book!  
Write powerful, clean and maintainable JavaScript.  
RRP \$11.95  
[Get the book free!](#)

  
**Jack Franklin**  
   
I'm a JavaScript and Ruby Developer working in London, focusing on tooling, ES2015 and ReactJS.

New books out now!  
  
Get practical advice to start your career in programming!

<https://www.sitepoint.com/organize-large-react-application/>

# Workshop



- Create a new, blank React project (or go ahead with your previous project)
- Identify and study the files discussed in this section
  - Run the project: `npm start`
- Replace some content in `App` component with your own code.
- Add some images.
- Create your own (new) component. Load it instead of `<App />`.
- Move component to a new `/components` folder.  
Make sure project still runs.
- Code example: `../100-helloworld`

