React Introduction

Component based UI Programming

Agenda

- What is MVC?
- Other Architectures
- What is React? Why?
- How to integrate?

Learn Objectives

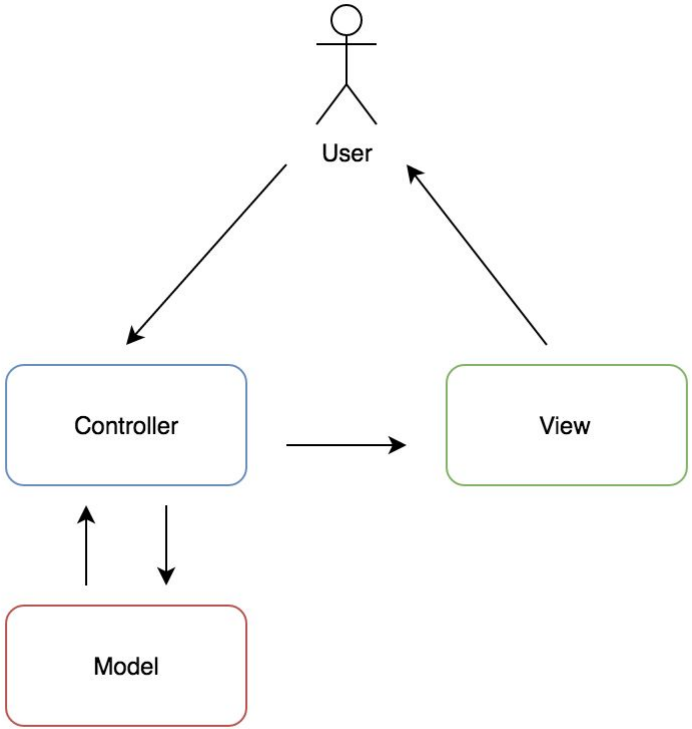
- You know UI Programming approaches
- You know ReactJs Fundamentals
- You know UI Design Pattern
- You know React CLI

UI Programming Approaches

Approaches

There are other approaches like

- MVC (Model-View-Controller)
  - MVM (Model-View-Model)
  - MVVM (Model-View-View-Model)
- Component Based



What is MVC (Model, View, Controller)?

MVC

The structure allows flexibility since responsibilities are clearly separated. This leads to

- better and easier code maintenance and reusability
- easier to coordinate in teams due to the separation
- ability to provide multiple views
- support for asynchronous implementations

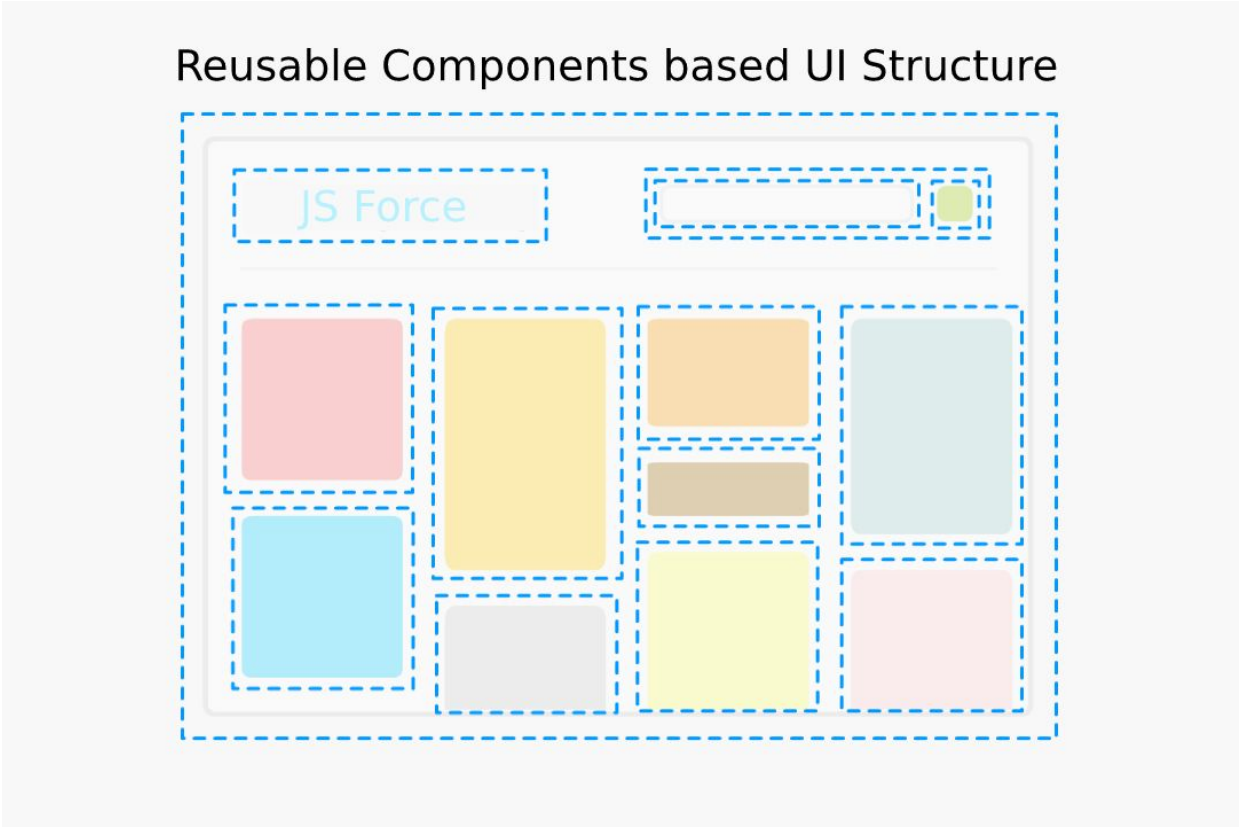
, but also to

- an increased complex setup process
- dependencies, i.e. changes in the model or controller affect the whole entity

Source:
https://medium.com/createdd-notes/understanding-mvc-architecture-with-react-6cd38e91fefd

Component based approach

- Components are reusable
- We used already that approach
  - HTML Slicing
  - Functional approach with Fragments
- In component based approach, there is a view tree (just like DOM)
- The view tree contains semantically unique and self isolated sections
- Basically a container of more html tags
- Not only tags, a component has its own javascript and css.
- Components could be nested and can build up another big components.



Sliced, isolated components

React

Component based approach

REACT

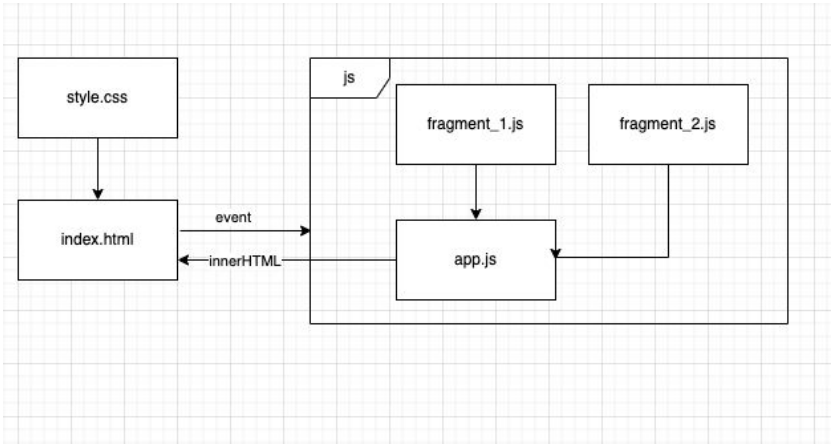
A component based approach

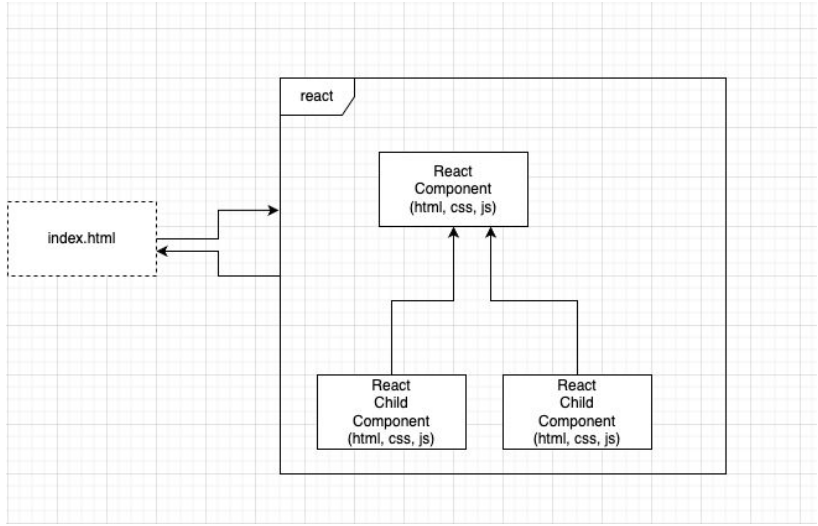
React is JavaScript library from Facebook, that is designed to create interactive UIs. The main features are that it’s

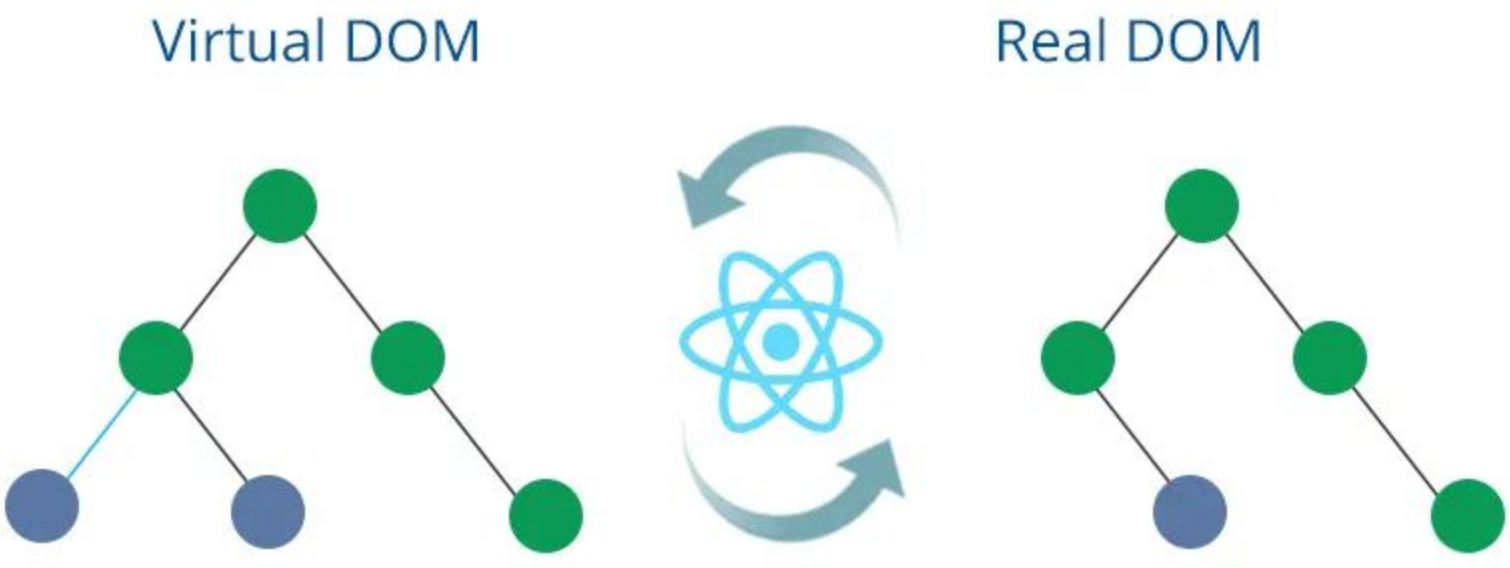
- **declarative**: Design different views for each state, which will be efficiently updated and re-rendered
- **component-based**: Build components, that manage their own state and structure them together into more complex UIs
- maintains an internal representation of the rendered UI (“**virtual DOM**”), that renders only the changed elements
- **Write once use everywhere approach**

2011 Facebook, 2012 Instagram

2013 MIT License


style.css
index.html
event
innerHTML
js
fragment_1.js
fragment_2.js
app.js


index.html
react
React
Component
(html, css, js)
React
Child
Component
(html, css, js)
React
Child
Component
(html, css, js)



Real & Virtual DOMs

Virtual DOM

React fundamental concepts

- React elements - JavaScript objects which represent browser HTML elements (h1, div, section …)
- Components - developer created React elements. Incapsulate some UI + functionality (NavBar, ImageUploader, Grid, PhotoGallery)
- JSX - xml-ish markup language for creating React elements, mix of html and javascript which is compiled to JavaScript. -> React.DOM.div({className: “people”})
- Virtual DOM - JavaScript tree of React elements in memory.

React's two programming approaches

- **Functional approach**
- OOP approach

```
import React from 'react';

const RecipeList = recipes => {
    return (
        <ul>
            {recipes.map(recipe => <li>{recipe.title}</li>)}
        </ul>
    )
}

export default RecipeList;
```

```
class App extends Component
{
    render()
    {
        return (<Hello name="Loy" />)
    };
}

class Hello extends Component
{
    render()
    {
        return (<div> Hello {this.props.name}!</div>);
    }
}
```

Some statistics

React
JavaScript library

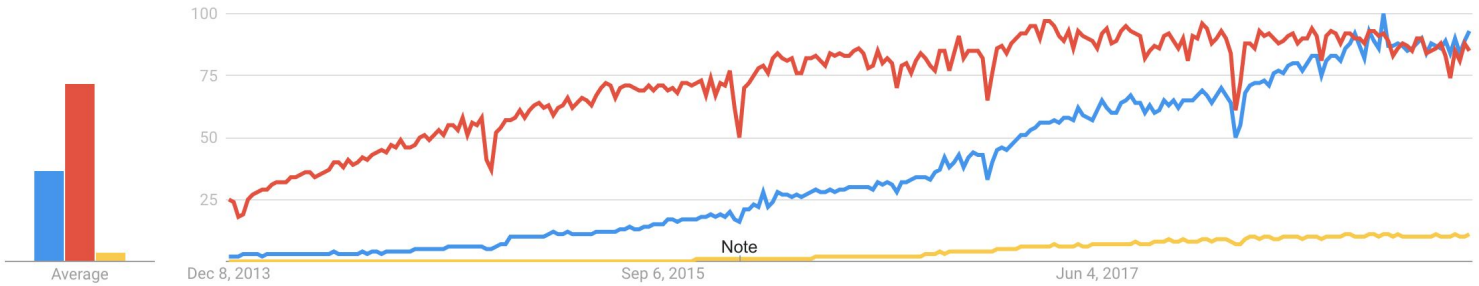
AngularJS
Topic

Vue.js
Topic

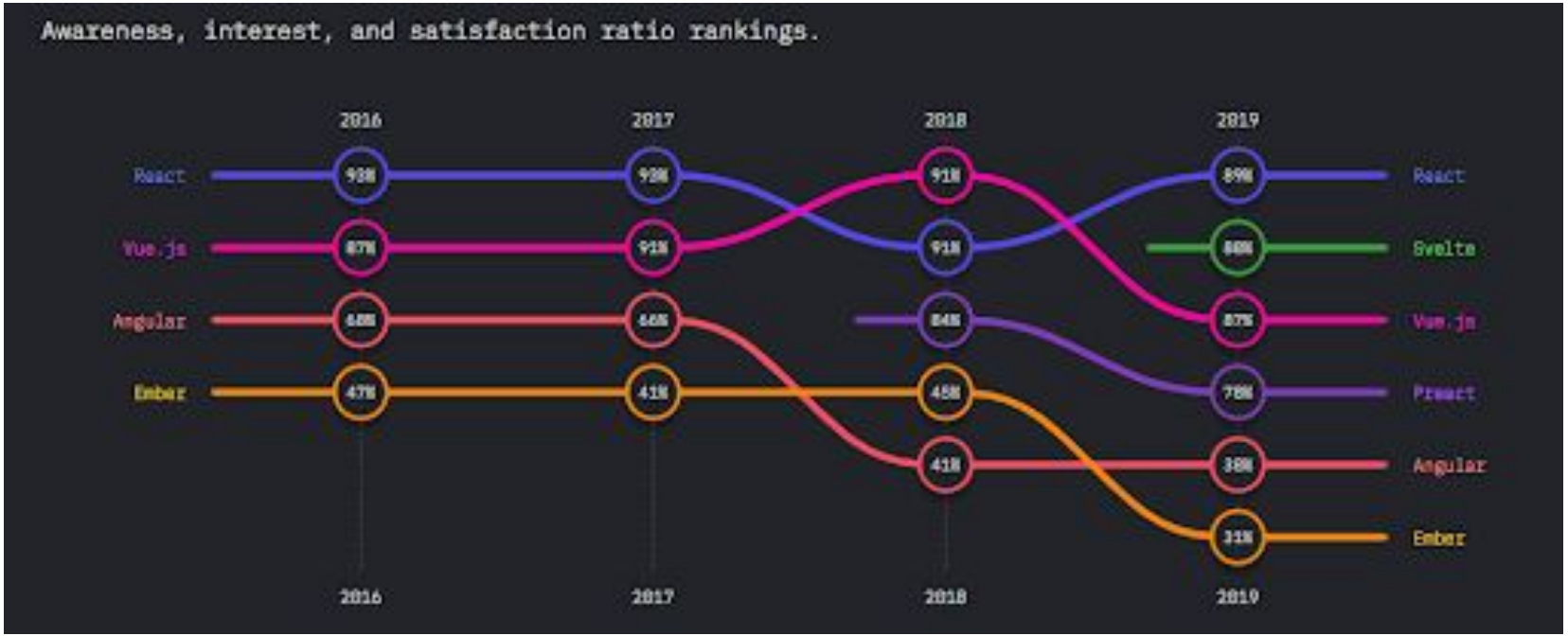
+ Add comparison

Worldwide | Past 5 years | All categories | Web Search

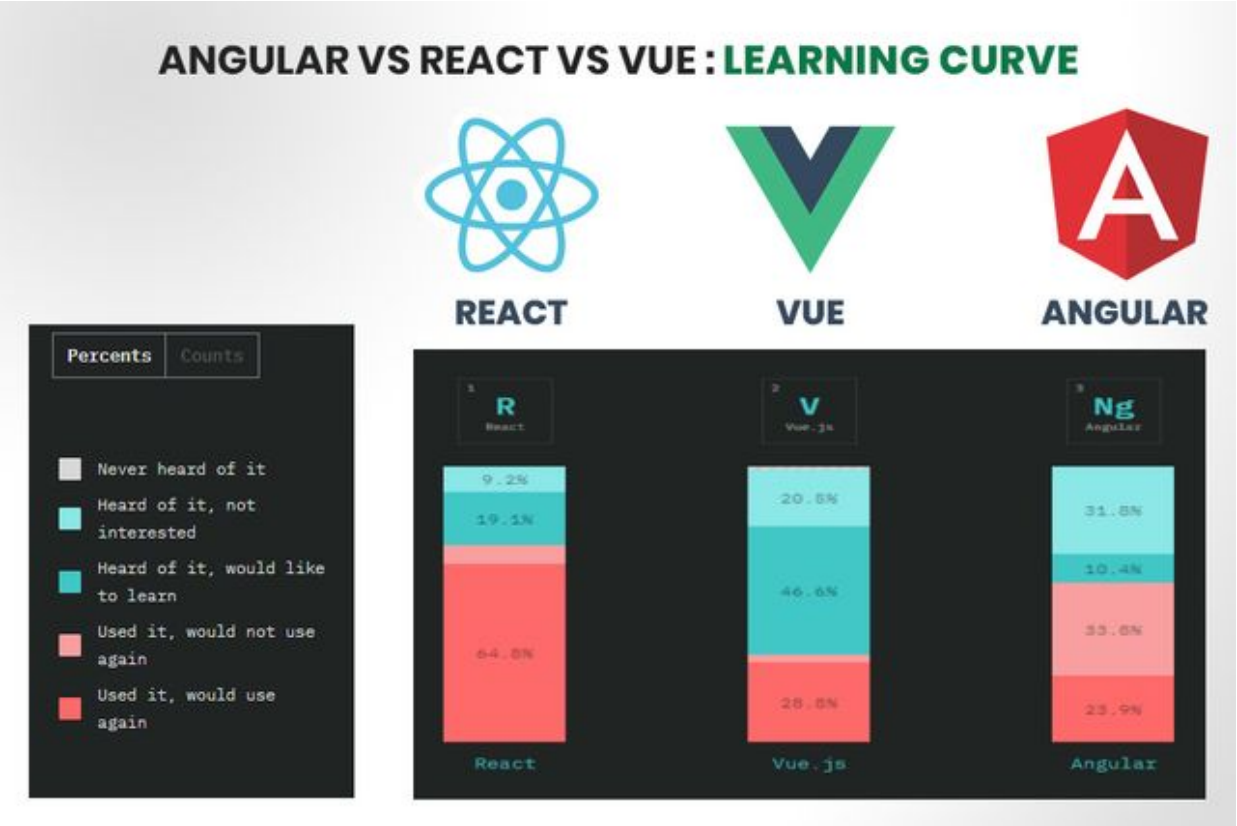
Interest over time



Tech trends



Overall interest



A comparison from a known developer survey

Most important concepts

```
function HelloMessage(props){
    return (
      <div>
        Hello {props.name}
      </div>
    );
}

ReactDOM.render(
  <HelloMessage name="Hi Coders!" />,
  document.getElementById('hello-example')
);
```

Hello Hi Coders!

A dead simple example from react itself

ReactDOM.render(...)

The joint point into DOM

It simplifies using the DOM API.

It looks like html but this funny tag syntax is neither a string nor HTML

```
const element = (
  <h1 className="greeting">
    Hello, world!
  </h1>
);
```

```
const element = React.createElement(
  'h1',
  {className: 'greeting'},
  'Hello, world!'
);
```

Render component on the page

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

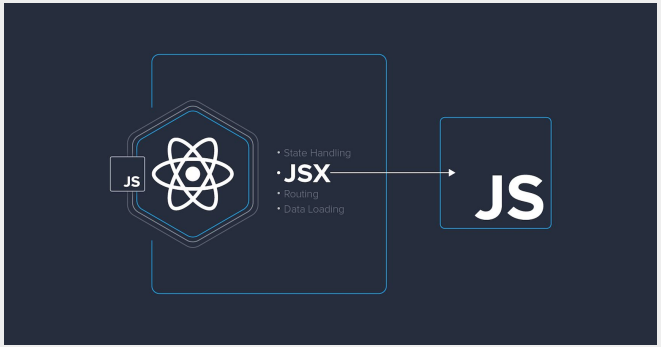
```
<!DOCTYPE html>
<html>
<head>...</head>
<body>
    <div id="root"></div>
</body>
</html>
```

JSX

Abstract using DOM API

It simplifies using the DOM API.

It looks like html but this funny tag syntax is neither a string nor HTML




```
const element = (
  <div>
    <h1>Hello!</h1>
    <h2>Good to see you here.</h2>
  </div>
);
```

Function

Functions are components

- In react, each component is created by using a function.
- A function (component) returns a jsx (html fragment)
- Component functions are to capitalize
- Functions could use other components just like HTML TAGs

JS index.js > ...

```
1
2   function Koltuk(){
3       return (
4         <div>
5           <h1>koltugum ben</h1>
6         </div>
7       );
8   }
9
10  function OturmaOdasi(props){
11      return (
12        <section>
13          <h1>oturma odasi</h1>
14          <Koltuk/>
15          <Koltuk/>
16        </section>
17      );
18  }
19
20  ReactDOM.render(
21    <OturmaOdasi name="Hi Coders!" />,
22    document.getElementById('ev-temeli')
23  );
24
```

A simple example for nested components

Props

How to pass values to a component

- Each component is a HTML TAG as used in another component
- Since a component is a function, it can gets arguments/parameters
- The parameters are passed down through using html attributes

index.js > ...

```
1
2   function Araba(props){
3       return (
4         <section>
5           <h1>Arabayim ben</h1>
6           <Koltuk/>
7           <Koltuk/>
8           <Direksiyon/>
9           <Gaz hiz={props.hiz}/>
10          <Fren/>
11          <BenzinDeposu tank={props.benzin}/>
12        </section>
13      );
14  }
15
16  ReactDOM.render(
17    <Araba benzin="50L" hiz="100km/h"/>,
18    document.getElementById('otoban')
19  );
20
```

How to pass parameters to a component.

No more use of innerHTML!

installation

How to integrate react?

1. Like we have done it before with CDN or similar
2. Using node and npm libraries

1 is suitable for small projects/prototypes, use second everywhere where the project is even small.

Use libraries from CDN

```
  <!-- Note: when deploying, replace "development.js" with "production.min.js". -->

  <script src="https://unpkg.com/react@16/umd/react.development.js"
crossorigin></script>

  <script src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"
crossorigin></script>

<script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
```

By using NPM

Steps to begin react programming

- npm install -g create-react-app
- npx create-react-app my-app
- cd my-app
- npm start

File Structure

1. Grouping by file type
2. Grouping by features or routes

Rules

- React doesn’t have opinions on how you put files into folders
- Avoid too much nesting
- Don’t overthink it just start then refactor

questions?