Școala
informală
de IT

React JS

Agenda

- What is React JS
- Components
- JSX
- React Ecosystem
- How to create a React app
- Props
- State

React JS - What it is and why it’s useful

React JS

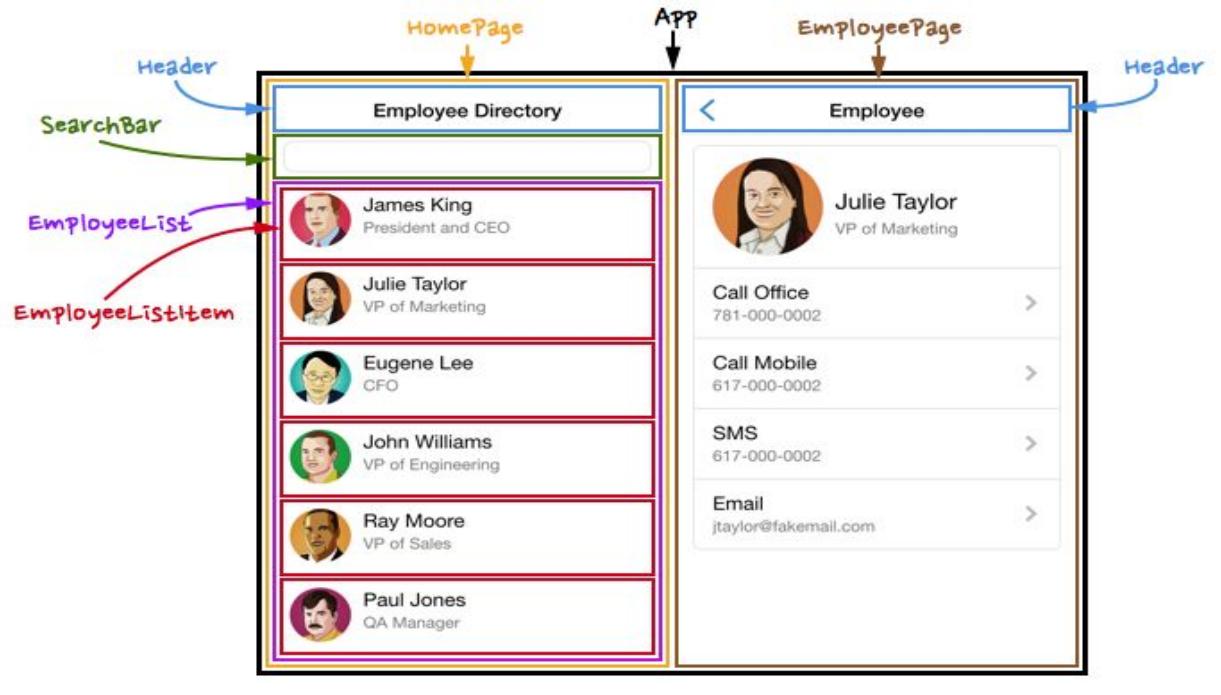
- = a Javascript library for building user interfaces, easier and faster.
- 3 main benefits of learning React:
  - React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right *components* when your data changes. Declarative views make your code more predictable and easier to debug.
  - Build encapsulated *components* that manage their own state, then compose them to make complex UIs.
  - React can also render on the server using Node and power mobile apps using React Native

Components

Components

- let you split the UI into *independent*, *reusable pieces*, and think about each piece in isolation
- Conceptually, components are like *JavaScript functions*
  - They accept arbitrary inputs (called "props") and
  - return React elements describing what should appear on the screen
- Can be defined as functions or classes. They are the same from the React point of view.
- Component's name should start with an uppercase letter

Components


HomePage
App
EmployeePage
Header
SearchBar
EmployeeList
EmployeeListItem
Header
Employee Directory
James King
President and CEO
Julie Taylor
VP of Marketing
Eugene Lee
CFO
John Williams
VP of Engineering
Ray Moore
VP of Sales
Paul Jones
QA Manager
Employee
Julie Taylor
VP of Marketing
Call Office
781-000-0002
Call Mobile
617-000-0002
SMS
617-000-0002
Email
jtaylor@fakemail.com

Hands-on

- Let’s use React as we would use a regular library
- Requirement: Display “Hello World” in a div with id “root”

```
const element = React.createElement(
    'h1', {},
    'Hello, world!’
);
ReactDOM.render(element, document.getElementById('root'));
```

Hands-on

- **2 libraries:**
  - `<script src="https://cdnjs.cloudflare.com/ajax/libs/react/15.4.2/react.js"></script>`
  - `<script src="https://cdnjs.cloudflare.com/ajax/libs/react/15.4.2/react-dom.js"></script>`

Hands-on

- **One more library:**
  - `<script src="https://cdnjs.cloudflare.com/ajax/libs/babel-standalone/6.21.1/babel.min.js"></script>`

```
ReactDOM.render(
    <h1>Hello, world!</h1>,
    document.getElementById('root')
);
```

JSX

JSX

- “The funny syntax” that is neither JS nor HTML
- It is a syntax extension to JavaScript
- used with React to describe what the UI should look like
- Looks like a template language, but it comes with the full power of JavaScript
- https://reactjs.org/docs/introducing-jsx.html

Hands on

- Let’s create a timer component together
- Display the current time below the “Hello World” component
- Use functional components syntax
- Install React Developer Tools

React Ecosystem and how to create a React app

React Ecosystem

1. Javascript Modules
2. Node Package Manager (npm)
3. Webpack
   - Creates a tree of dependencies
   - Brings all files as IIFEs and puts them in a single file
   - Pre and post “compilation” (web pack plugins)
4. Babel
   - Transpiles code from ES6 (and any future JS feature) into standard (supported by browsers) JS
   - Converts JSX to JS

How to create a React App (easily)

- **Create React App package (getting started)**
  - **Abstracts the basic setup so you can start writing your components like in a regular project**
- **Code Sandbox (link)**
  - **Online tool for starting up and sharing React code (project) examples**

Hands-on

- Use Create React App
  - Create the timer using create react app
  - Create a separate component for the actual timer

Props & State

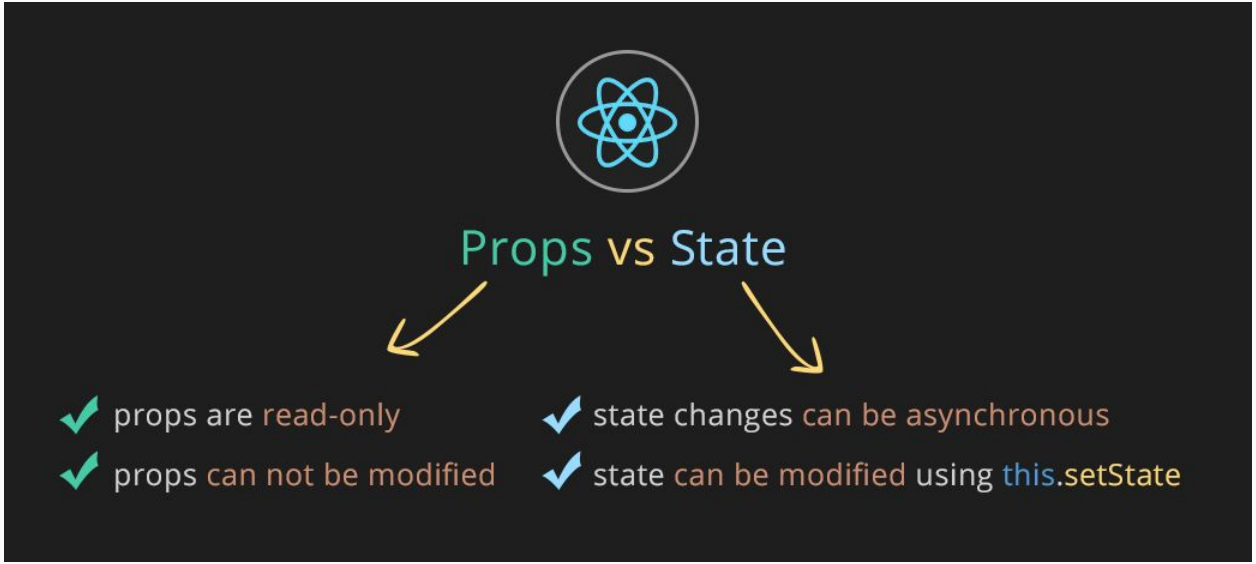
(More on) Props

- Most components can be customized when they are created, with different parameters.
- These creation parameters are called props.
- Props are read-only
  - A component must never modify its own props!
  - In other words, all React components must act like pure functions with respect to their props.
- Hands-on
  - Change “Hello World” to display “Hello {name}”

State

- **Props cannot be changed, but the data on the UI is dynamic (it changes). To keep this data, use component's internal state**
  - **Internal => It is visible and can only be updated by that component**
- **Try to keep a component's internal state as simple as possible, avoid having a deep structure stored on it**
- **Initial state - this is set in *constructor* function as the initial state of your component**
- **To update the state, use setState - a predefined React method**
  - **setState is an async function!**
  - **Never update this.state directly!**

Props vs State



Component Lifecycle

Component Lifecycle

Each component has several "lifecycle methods" that you can override to run code at particular times in the process:

- **Methods prefixed with "will" are called *right before* something happens**
- **Methods prefixed with "did" are called *right after* something happens**

Official docs: https://reactjs.org/docs/react-component.html

Hands on

- Let’s reuse the timer component created together
- Use functional components and class components syntax

Functional vs class components

You can convert a functional component to a class in five steps:

1. **Create an ES6 class, with the same name, that extends React.Component.**
2. **Add a single empty method to it called render().**
3. **Move the body of the function into the render() method.**
4. **Replace props with this.props in the render() body.**
5. **Delete the remaining empty function declaration.**

Handling events

Handling events

- Very similar to handling events on DOM elements
- Syntactic differences:
  - React events are named using camelCase, rather than lowercase
  - With JSX you pass a function as the event handler, rather than a string

Docs: https://reactjs.org/docs/handling-events.html

Hands on

- Let’s reuse the timer component created together
- Add a start and a stop button for the timer

Resources

https://reactjs.org

https://medium.freecodecamp.org/javascript-modules-a-beginner-s-guide-783f7d7a5fcc

https://docs.npmjs.com/getting-started/what-is-npm

https://medium.freecodecamp.org/all-the-fundamental-react-js-concepts-jammed-into-this-single-medium-article-c83f9b53eac2

https://www.codecademy.com/learn/react-101

https://reactjs.org/tutorial/tutorial.html