React Introduction

hello world

Lesson Plan

- What is React
- Hello world with Vanilla JS
- Hello World with react
- React.createElement
- JSX
- VirtualDOM
- Architecture

Lesson Summary

https://www.nerdeez.com/articles/react/introduction

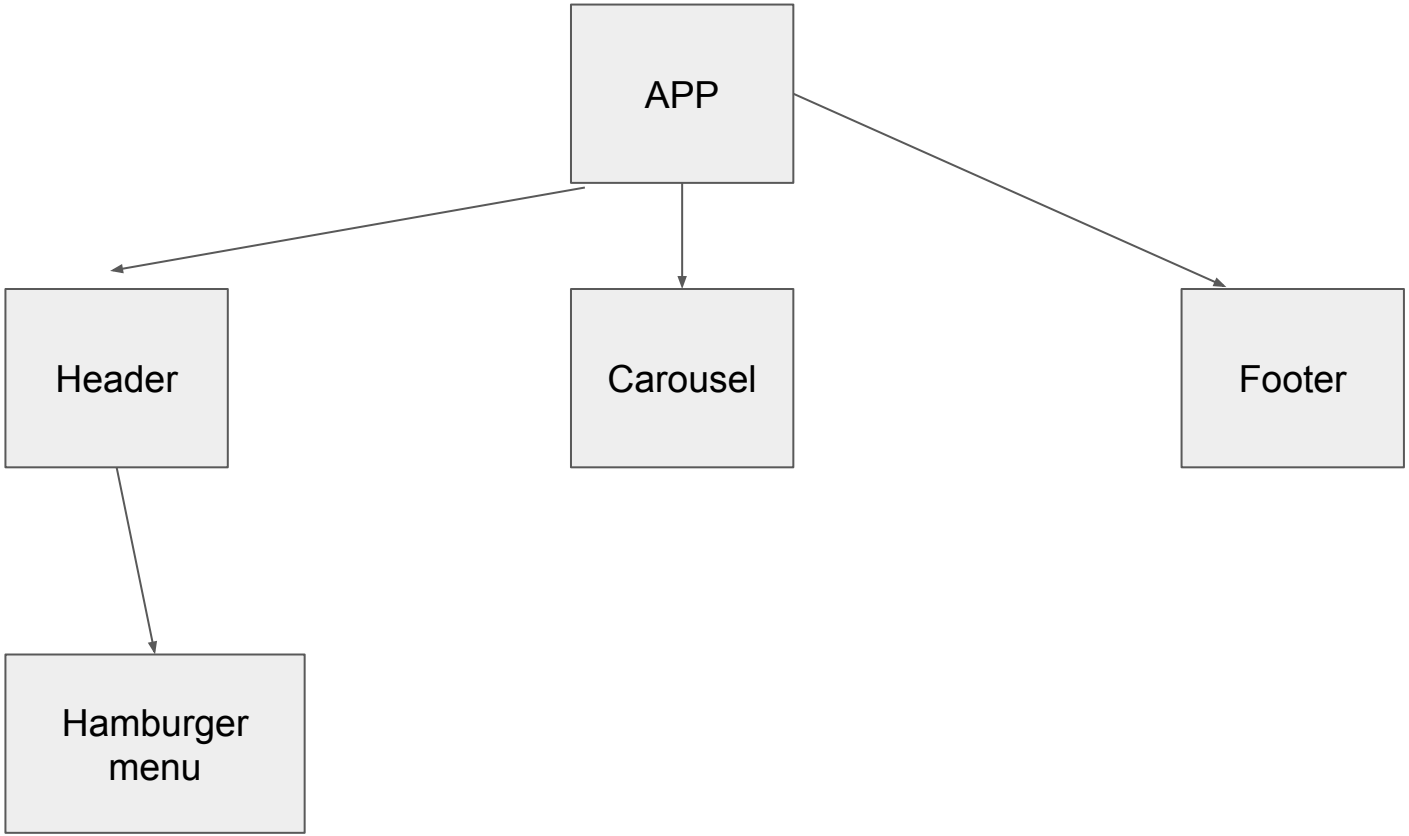
What is React

- UI Library - the V in an MVC
- We create block of ui we call react components
- Open sourced and maintained by facebook
- Called React cause it reacts to components change and knows when to redraw them
- Unopinionated and can be combined in a non react page
- When combined with a state library like redux can create high performance scalable web application
- React has been developed since 2011, open sourced in 2013 and is used in production in many companies like: facebook, wix, instagram etc.

React Architecture

- Lets say we want to create a page in React that looks like the page in the url:
  - https://www.bugeez.io
- When building with react we are starting to divide our application to block of UI
- we decide on the blocks by thinking about separation of concerns
  - a block of ui is addressing a certain defined problem
  - minimum connection to the external world
  - optimal for reuse in other places
- so the bugeez page might have the following block of UI components
  - header
  - footer
  - carousel
  - hamburger menu
- A React application is a wrapped in a single UI block that contains the entire app

React Architecture



Hello world with vanilla JS

- First lets try to create an app that displays an hello world with JS without using React

Hello World with React

- Lets build an hello world with React
- To use React we will have to install two libraries
  - React - `https://unpkg.com/react@16/umd/react.production.min.js`
  - ReactDOM - `https://unpkg.com/react-dom@16/umd/react-dom.production.min.js`
- With React library we can build react components
- with React-dom we can attach our components to the DOM
- Loading the scripts will expose two globals
  - React
  - ReactDOM
- React.createElement will create a react component
- ReactDOM.render - will place the component on the dom

React.createElement

- can create nested tags - lets place a span inside our h1
- can get a function which returns React.createElement
- can get a class which extends React.Component
- But it’s still very clumsy to write nested React.createElement, we need an easier way to create those elements

JSX

- JavaScript XML
- an extension to JS
- browsers do not understand it
- we need a tool that will convert that syntax to React.createElement
- we use Babel for this
  - demo: https://babeljs.io/en/repl
- place the following script in the head
  - `https://cdnjs.cloudflare.com/ajax/libs/babel-standalone/6.26.0/babel.js`
  - script with type **text/babel** will be transpiled
- Lets change our app to work with JSX instead of the React.createElement
- notice that you will have to have React in the scope to use it since the code is transpiled to React.createElement

VirtualDOM

- DOM manipulation is expensive
- React creates a virtual dom tree with the React.createElement we are creating
- react will know when the components need to be redrawn and after that they will compare the current state with what the dom contains and perform the necessary changes.

Summary

- Now that we are working with React we will have a single component wrapping our entire application
- we split our apps to ui components called React components according to separation of concerns.
- React will make sure to update the DOM when our components change.
- to practice what we learned try and create an hello world app yourself.