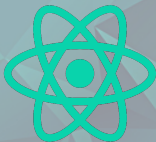


REACT BASICS

Bernardo Cuteri



- Open source framework for front-end development
- Maintained by Facebook and a community of individual developers and companies
- Mainly used for single page applications and mobile applications
- First released in 2013

SETUP

- Prerequisites: Nodejs and NPM

create-react-app commandline tool: `npm install -g create-react-app`

Create a React app with commandline tool

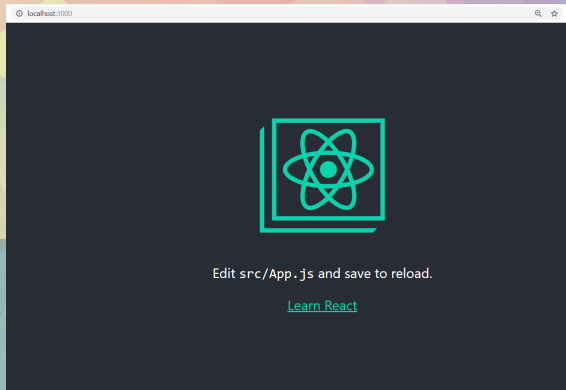
```
$ npx create-react-app my-app
```

Run the app:

```
$ npm start
```

A development server will be started at port 3000 (you can access it by typing `localhost:3000` in a browser).

Running the app gives the following result



REACT HELLO WORLD

An Hello World app

```
src > JS index.js > ...  
1   import React from 'react';  
2   import ReactDOM from 'react-dom';  
3  
4   const hello = <h1>Hello World!</h1>  
5  
6   ReactDOM.render(hello, document.getElementById('root'));  
7
```

Try to modify the example.. the application reloads automatically in the browser!

REACT VIRTUAL DOM

React uses a virtual DOM maintained in memory

- Document manipulation is done in the virtual DOM
- Changes are applied in the browser DOM only when needed
- React only changes what needs to be changed

React uses ES6

- ES6 stands for ECMAScript 6
- ES has been created to standardize JavaScript
- First published in 2015
- A prerequisite for React Developers
- Some constructs include: Classes, Arrow Functions and Variables (let, const, var)

A class is a type of function instantiated by the **class** keyword

- Classes have properties assigned in a **constructor()**
- An object is an instance of a class and is instantiated by the **new** keyword
- Additional class methods can be declared
- Inheritance is supported (**extends** keyword): **super()** refers to the parent class

ARROW FUNCTIONS

- Introduced in ES6
- More compact way of writing functions

Arrow functions interpret the **this** keyword differently from regular functions

- In an arrow function, **this** refers to the object that defines the arrow function
- In a regular function, **this** refers to the object that called the regular function

An example of arrow function that takes a parameter and logs it in the console:

```
printMessage = (msg) => {  
  console.log(msg);  
}
```

VARIABLES

Three ways of defining variables:

- **var**
 - outside of a function -> global scope
 - inside of a function -> function scope
 - inside of a block -> still function scope
- **let** -> block scope
- **const** -> used for constants

RENDERING HTML IN REACT

React is concerned with rendering HTML in web pages

- **ReactDOM.render()** -> used to render HTML code in an HTML element
- **Root Node** -> an HTML element where React performs rendering (id/name not important)

```
src > JS index.js > ...
```

```
1  import React from 'react';
2  import ReactDOM from 'react-dom';
3
4  const hello = <h1>Hello World!</h1>
5
6  ReactDOM.render(hello, document.getElementById('root'));
7
```

JSX

- Stands for JavaScript XML
- Allows to write HTML in JS code
- The preferred way of writing HTML code in React
- Curly brackets are used to write ES expressions
- Brackets for multi-line code
- Only one top level element
- Strict XML syntax: empty elements must be closed

```
const message = (  
  <div>  
    <h1>1 + 1 = {1 + 1}</h1>  
  </div>  
)  
ReactDOM.render(message, document.getElementById('root'));
```

REACT COMPONENTS

- Reusable code
- Work in isolation
- Return HTML

REACT CLASS COMPONENTS

- Extend **React.Component** base class
- Require a **render()** method
- **render()** returns HTML (uses JSX)

```
src > components > JS Person.js > ...
```

```
1  import React from 'react';
2  class Person extends React.Component {
3    |   surname = "Pasticcio";
4    |   name = "Ciccio";
5    |   render() {
6    |     |   return <h1>I am {this.name} {this.surname} </h1>
7    |     |   }
8    |   }
9  export default Person;
```

- Components are used in the application using HTML syntax
- If there is a **constructor()** it will be invoked at component initialization
- Components can be nested (i.e. used in other components)

```
src > JS index.js
```

```
1  import React from 'react';
2  import ReactDOM from 'react-dom';
3  import Person from './components/Person';
4
5  ReactDOM.render(<Person/>, document.getElementById('root'));
6
```

REACT CLASS COMPONENTS FILES

- It can be ideal to put components into separate files
- Components files extension is **.js**
- They start by importing React
- They have to end with the statement `export default X;` where *X* is the name of the component class

REACT CLASS COMPONENTS STATE

- Components properties should be kept in an object called **state**
- When the state changes, the component re-renders affected parts
- **setState()** is used to change the state (or parts of it)

REACT PROPS

- Props are arguments passed into React components
- They are passed with the syntax of HTML attributes
- Are used to pass data from one component to another
- Props are accessed by the **props** object
- If a constructor exists, it has to take props as parameter and always pass them to the super class via **super()**



QUESTIONS??