REACT BASICS

Bernardo Cuteri

React

- 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
```

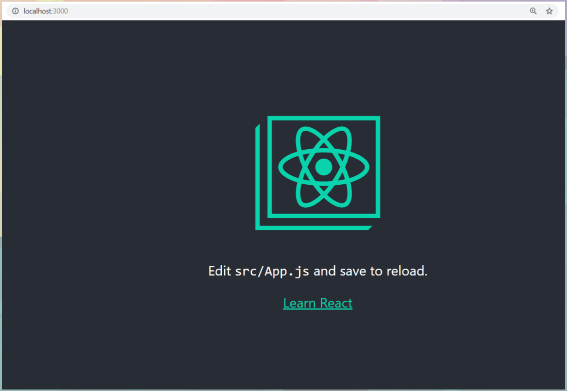
Run the app:

```
$ npm start
```

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

Setup (2)

Running the app gives the following result

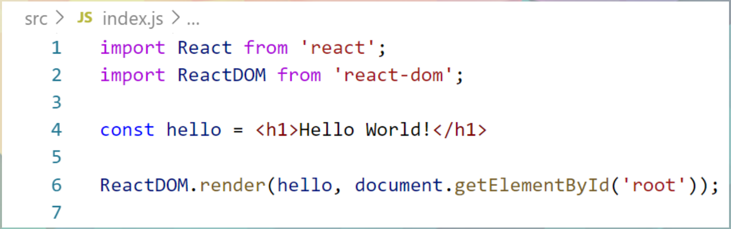


Edit `src/App.js` and save to reload.

Learn React

React Hello World

An Hello World app



src > JS index.js > ...

```
import React from 'react';
import ReactDOM from 'react-dom';

const hello = <h1>Hello World!</h1>

ReactDOM.render(hello, document.getElementById('root'));

```

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 ES6

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)

ES6 Classes

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

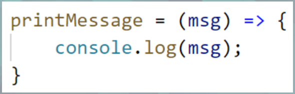
Arrow functions

- 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

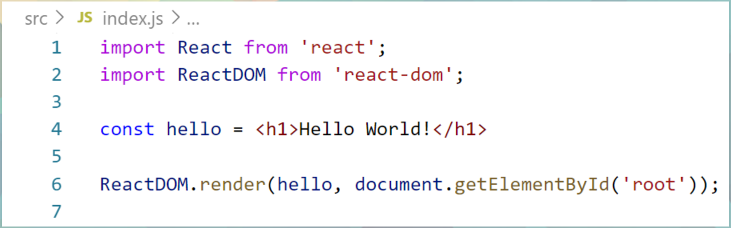
Three ways of defining variables:

- **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)

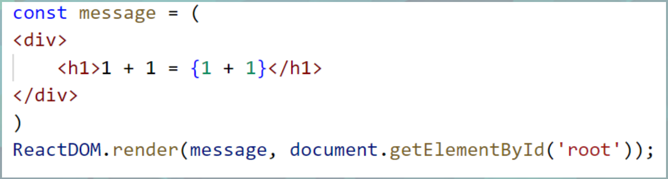


src > JS index.js > ...

```
import React from 'react';
import ReactDOM from 'react-dom';

const hello = <h1>Hello World!</h1>

ReactDOM.render(hello, document.getElementById('root'));

```

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



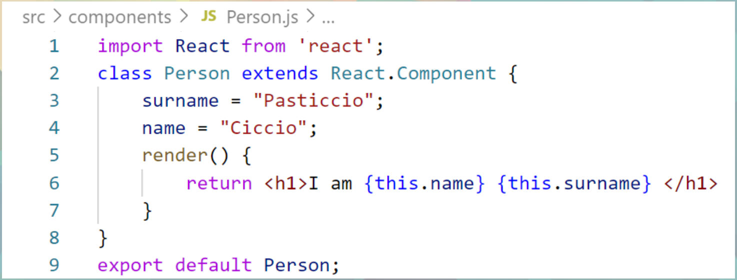
```
const message = (
<div>
    <h1>1 + 1 = {1 + 1}</h1>
</div>
)
ReactDOM.render(message, document.getElementById('root'));
```

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)

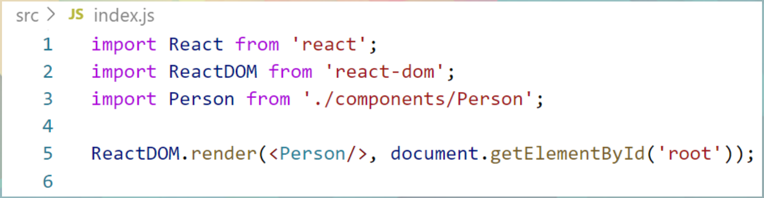


src > components > JS Person.js > ...

```
import React from 'react';
class Person extends React.Component {
    surname = "Pasticcio";
    name = "Ciccio";
    render() {
        return <h1>I am {this.name} {this.surname} </h1>
    }
}
export default Person;
```

React Class Components (2)

- Components are used in the application using HTML syntax



src > JS index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import Person from './components/Person';

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

```

React Class Components (3)

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

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)

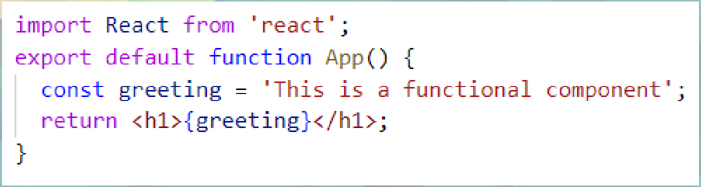
REACT PROPS

- 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()**
- React Props are read-only (i.e. you can not modify them)

React Functional Components

Functional components are components defined as JS functions

- the implemented function corresponds to the render method
- designed to implement stateless components
- in latest React versions can make use of state and lifecycle by using React hooks



```
import React from 'react';
export default function App() {
  const greeting = 'This is a functional component';
  return <h1>{greeting}</h1>;
}
```

QUESTIONS??