COESIONE ITALIA 21-27 PIEMONTE

Cofinanziato dall'Unione europea

REGIONE PIEMONTE

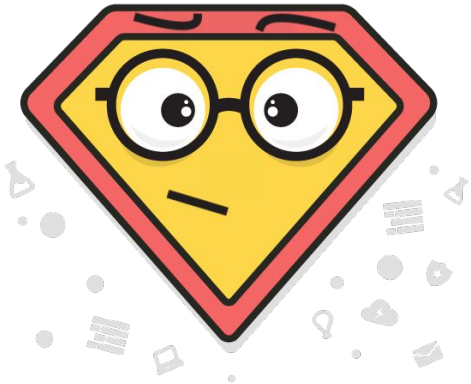
Programmazione Web

React

Davide Mantovani

synesthesia
the digital experience company

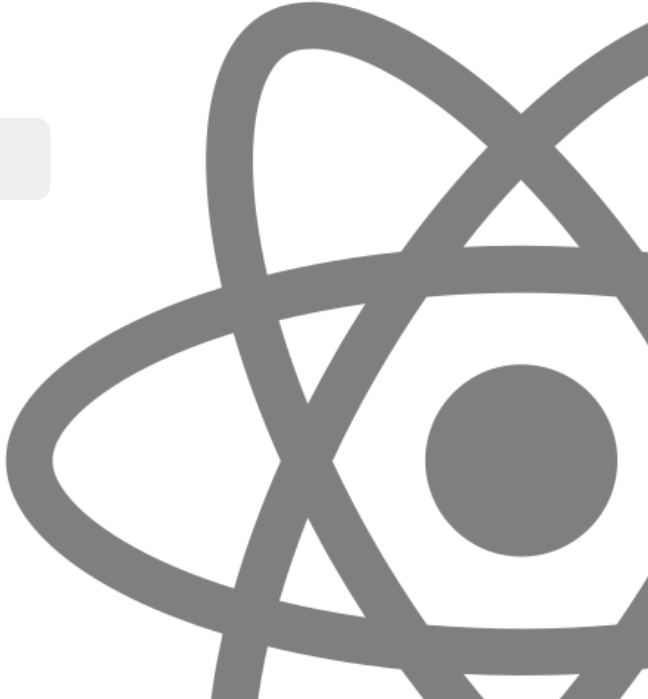
Intervento realizzato da ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE



React JS

From zero to n00b

But first, hands on!



IDE setup

Visual Studio Code is a lightweight but powerful source code editor that runs on the web or desktop and is available for Windows, macOS, and Linux. It includes out-of-the-box support for JavaScript, TypeScript, and Node.js, and offers a rich ecosystem of extensions for other languages and runtimes (e.g., C++, C#, Java, Python, PHP, Go, .NET).

Web version → https://vscode.dev/

Desktop app → https://code.visualstudio.com/

Basic HTML page

https://gist.github.com/davesyn/8bd2dc059ae3f8ceddb59ebd50ff7f5d

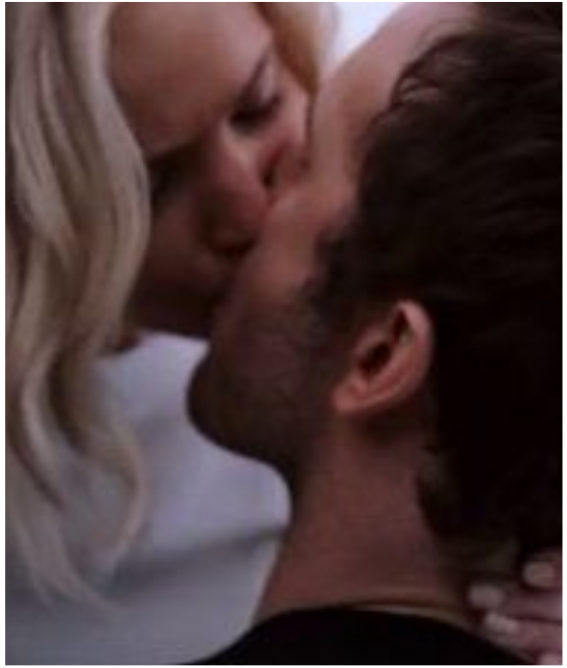
Basic STATIC React HTML page

https://gist.github.com/davesyn/87142b95f3ae8ca5719fc9518b8b1531 (not for production)

Basic INTERACTIVE React HTML page

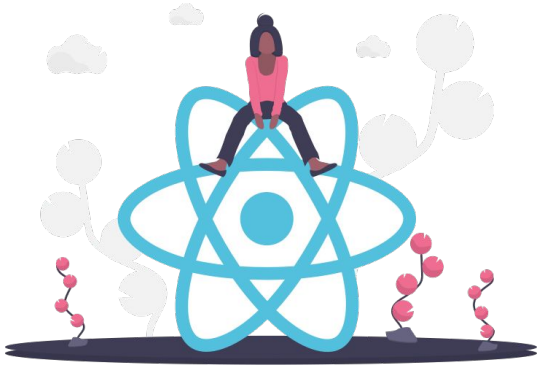
https://gist.github.com/davesyn/c845476554bff9ebdeb9f93259454f53 (not for production)

What Is React?

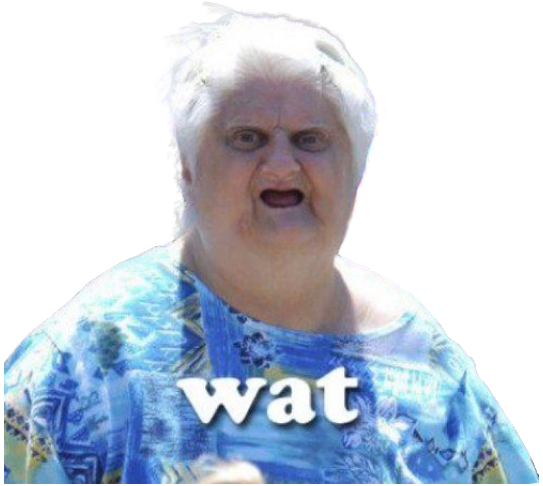


What Is React?

- Library/UI framework released by Facebook in 2013
- ReactJS takes care to render JSX into the DOM
- Core programming is all done in pure Javascript/Typescript
- Supports all EcmaScript latest features



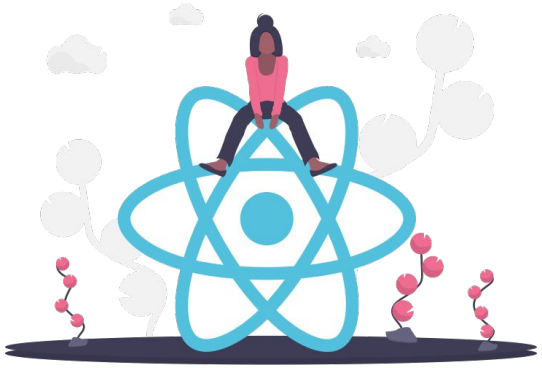
WAIT WAT... a bit of glossary



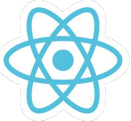
- **UI:** User Interface
- **JSX:** Syntax to create HTML elements in JavaScript
- **DOM:** HTML document model made of objects
- **EcmaScript:** JavaScript standard meant to ensure the interoperability of Web apps across different Web browsers

Again... What Is React?

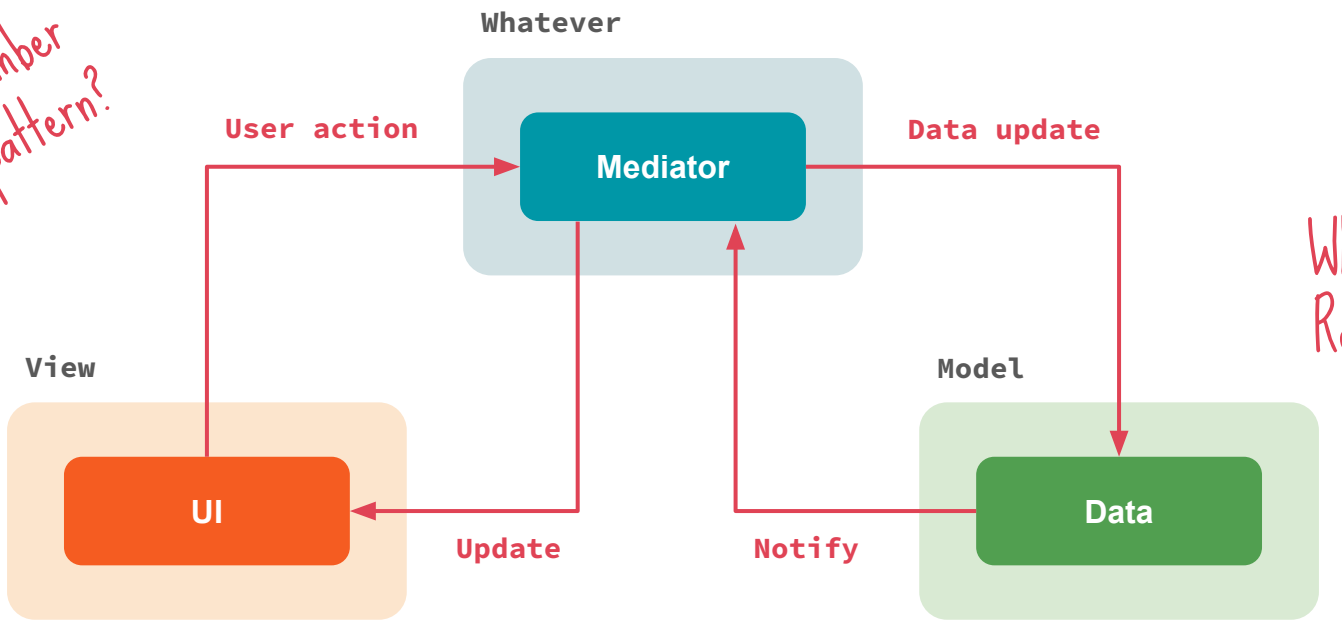
- **Library/UI framework** released by **Facebook** in **2013**
- ReactJS takes care to render **JSX** into the **DOM**
- Core programming is all done in pure **Javascript/Typescript**
- Supports all **EcmaScript** latest features



Again... What Is React?

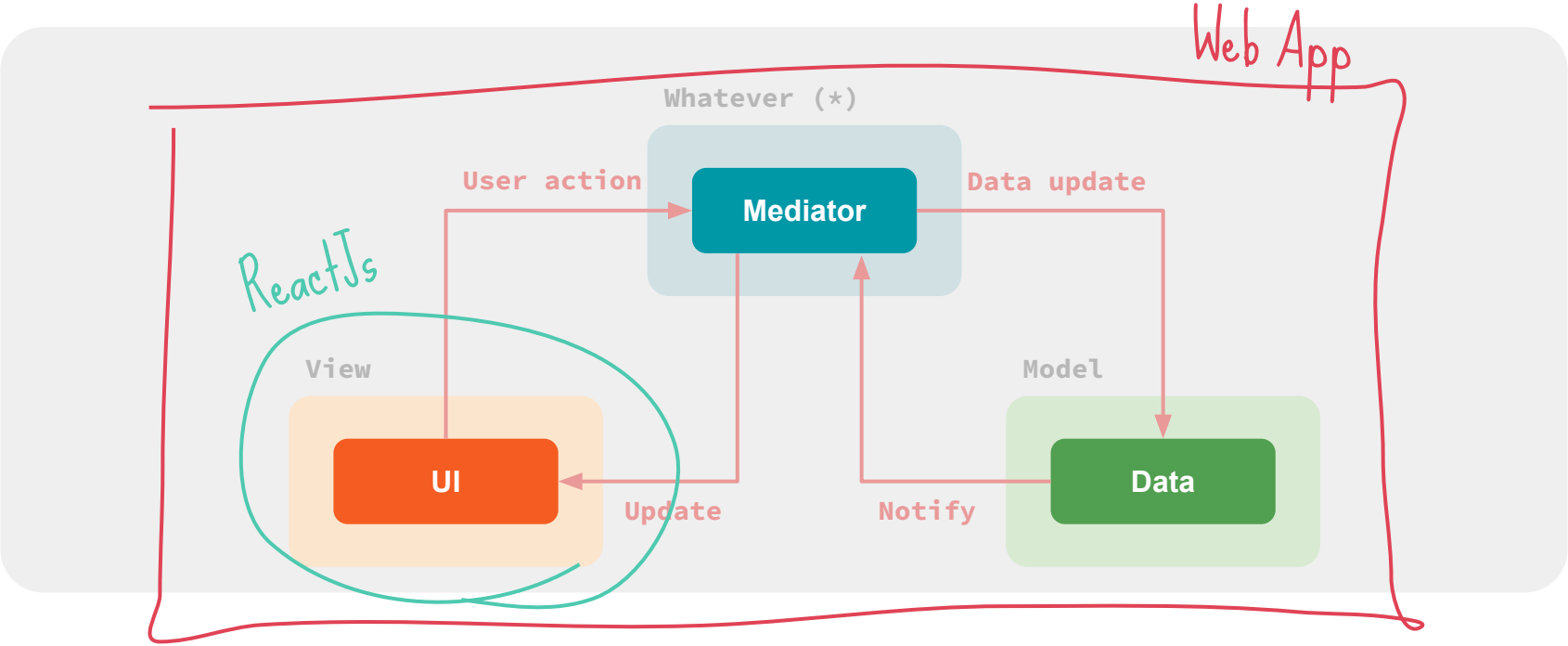


Do you remember the MV* pattern?



Where's React?

Again... What Is React?



Again... What Is React?

- Has an enormous **community** all around the Net
- Used by many **companies**

ES: not your grandma JS

```
array = [1, 2, 3, 4, 5];

for(i = 0; i < array.length; i++) {
    array[i] = array[i] + 1;
} // normal js

array = array.map(function (v) { return v + 1; }); // es5

array = array.map(v => v + 1); // es6
```

Map array and arrow functions

ES: not your grandma JS

```
curr = [ 4, 5, 6 ];
prev = [ 1, 2, 3 ].concat(curr); // es5

prev = [ 1, 2, 3, ...curr]; // es6
```

Array spread operators

```
user = { id: 1, name: "Dave" };
flags = { premium: true };

user = { ...user, ...flags }; // es6
```

Object spread operators

ES: not your grandma JS

```
message = "Hello " + customer.name + "!"; // normal js

message = `Hello ${customer.name}!`; // es6
```

String templates

```
x = 0; y = 0;

obj = { x: x, y: y }; // es5

obj = { x, y }; // es6
```

Property shorthand

ES: not your grandma JS

ecma INTERNATIONAL

```
array = [ 1, 2, 3 ];

var a = array[0], b = array[2]; // normal js
let [ a, , b ] = array // es9

obj = { id: 1, name: "Dave" };

var id = obj.id, name = obj.name; // normal js
let { id, name, premium = true } = obj // es9
```

Destructuring

ES: not your grandma JS

```
var xmlHttp = new XMLHttpRequest();
xmlHttp.onreadystatechange = function() {
  console.log(xmlHttp.responseText);
};
xmlHttp.open("GET", url, true);
xmlHttp.send(null); // normal js

fetch(url)
  .then(function(res) { return res.json() })
  .then(function(data) { console.log(data) }); // es5

let res = await fetch(url);
console.log(await res.json()); // es8
```

Promise and
async/await

Can I use...?

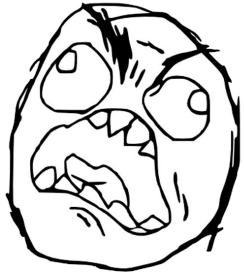
ecma INTERNATIONAL

ECMAScript 2015 (ES6)

Support for the ECMAScript 2015 specification. Features include Promises, Modules, Classes, Template Literals, Arrow Functions, Let and Const, Default Parameters, Generators, Destructuring Assignment, Rest & Spread, Map/Set & WeakMap/WeakSet and many more.

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Chrome for Android	Firefox for Android
	[2] 12-14	2-5	4-20	3.1-7	3.2-6.1		
	[2 3] 15-18	6-53	21-50	7.1-9.1	7-9.3		
6-10	[2] 79-84	[2] 54-80	[2] 51-84	10-13.1	10-13.7		
[1 2] 11	[2] 85	[2] 81	[2] 85	14	14.0	[2] 85	[2] 79
		[2] 82-83	[2] 86-88	TP			

→ caniuse.com/



Filling the dots

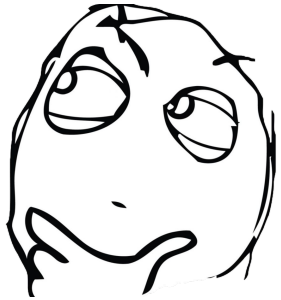
ecma INTERNATIONAL

BABEL

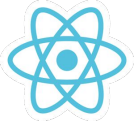
```
// Input: es6 arrow function
[1, 2, 3].map((n) => n + 1);

// Babel Output: es5 equivalent
[1, 2, 3].map(function(n) {
  return n + 1;
});
```

<Polyfills>

```
function isArray(a) {
  if (Array.isArray) {
    // if available
    return Array.isArray(a);
  }
  // polyfilling instead
  return Object.prototype.toString
    .call(a) === '[object Array]';
}
```



Speaking about ReactJS



- It's (mostly) just a cooler version of **HTML+JS**
- Everything is a **component**
- It's all **state** and **props**
- What you **render** is what you **get**

A cooler version of HTML+JS - example 1

```
<button onclick="javascript:alert('Saving');">
  Save
</button>
```

HTML

```
<button onClick={() => alert('Saving');}>
  Save
</button>
```

JSX

Regular button tag

A cooler version of HTML+JS - example 2

```
<div class="container" style="background-color: #eee; margin-top: 10px">
  <label for="name">Enter your name: </label>
  <input id="name" type="text" value=" "/>
  <button onclick="javascript:alert('Saving');">Save</button>
</div>
<p>This page is useless.</p> <!-- This is a comment -->
```

HTML

```
<>
  <div className="container" style={{ backgroundColor: "#eee", marginTop: 10 }}>
    <label htmlFor="name">Enter your name: </label>
    <input id="name" type="text" defaultValue=" " />
    <button onClick={() => alert('Saving');}>Save</button>
  </div>
  <p>This page is useless.</p> {/* This is a comment */}
</>
```

JSX

Small page example

A cooler version of HTML+JS - example 3

```
<div>
  <h2>This is a card title</h2>
  <button onclick="javascript:alert();">Cool</button>
</div>
```

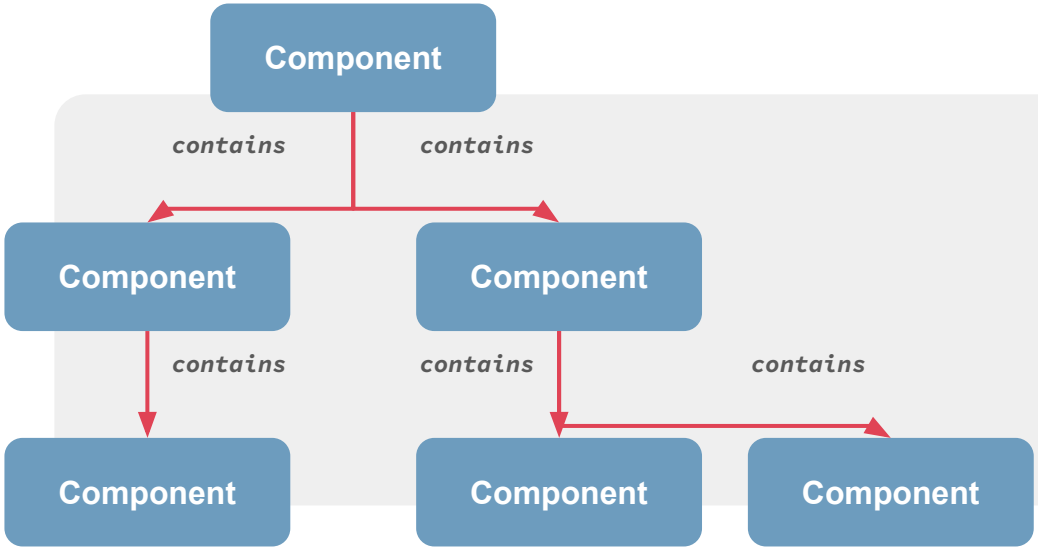
HTML

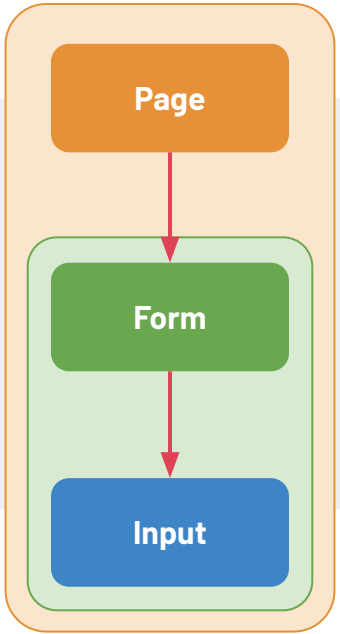
```
<MyCustomCard
  title={“This is a card title”}
  buttonAction={() => alert();}
  buttonLabel={“Cool”}
/>
```

JSX

Custom component!

Everything is a component





Everything is a component

```
// functional component
function Welcome(props) {
  return <h1>Hello, {props.name}</h1>;
}

// class component
class Welcome extends React.Component {
  render() {
    return <h1>Hello, {this.props.name}</h1>;
  }
}
```

Two ways to declare a <Welcome /> component

Functional and Class components

Everything is a component

```
// functional component with es6 arrow function
const Welcome = ({ name }) => <h1>Hello, {name}</h1>;

// class component
class Welcome extends React.Component {
  render() {
    const { name } = this.props; // es6 destructuring
    return <h1>Hello, {name}</h1>;
  }
}
```

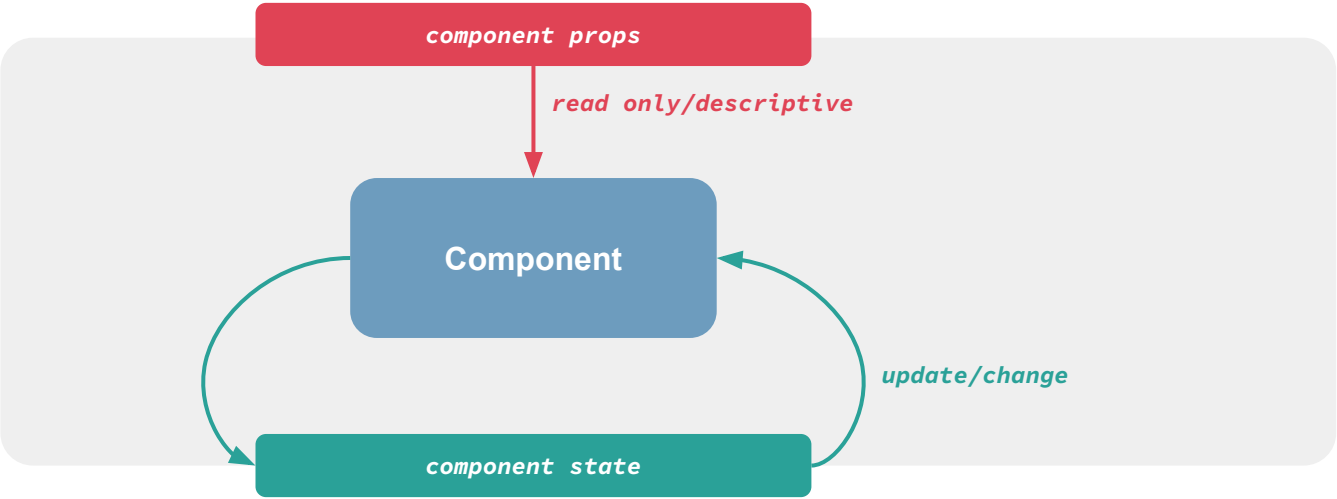
Functional and Class components (es6 way)

It's all state and props


```
// JSX custom component with “name”
const name = "Dave";

const element = <Welcome name={name} />;

// this works both for Class and Function Comps
```

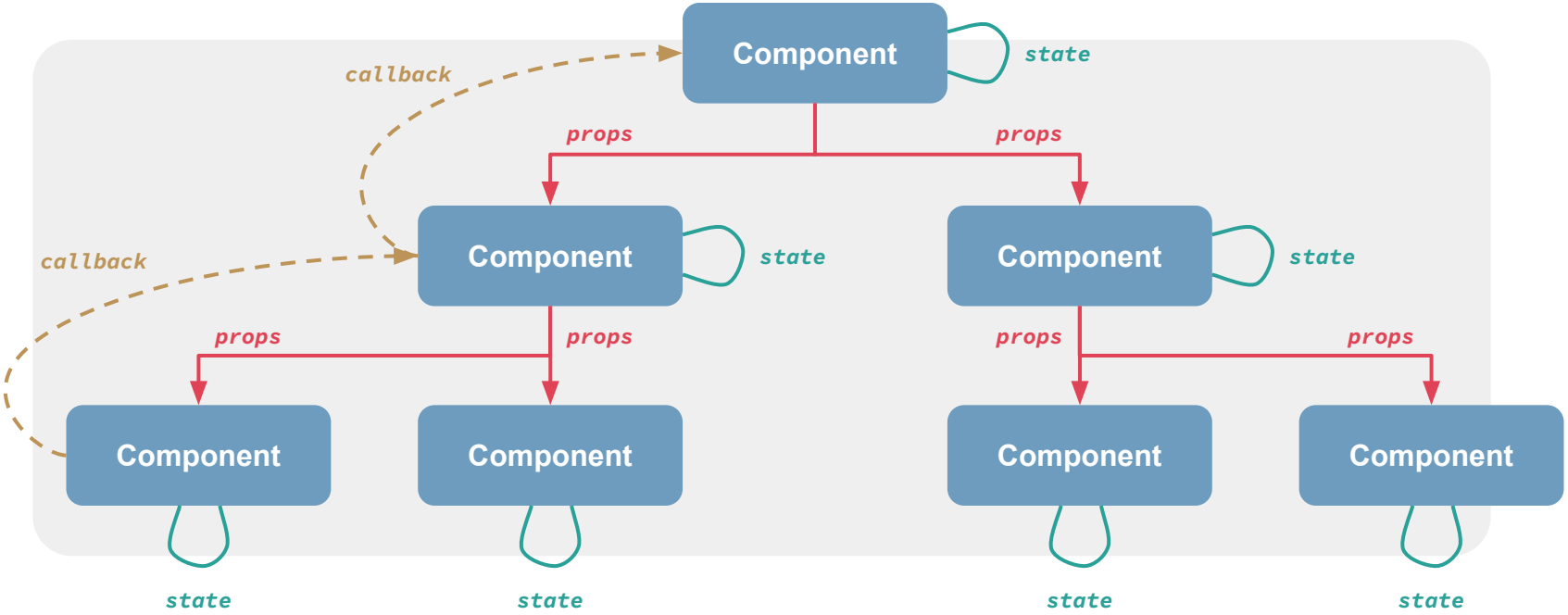
Pass props to a component

It's all state and props




```
this.setState({ key: value })

const [ value, setValue ] = useState(null)
```

It's all state and props



It's all state and props

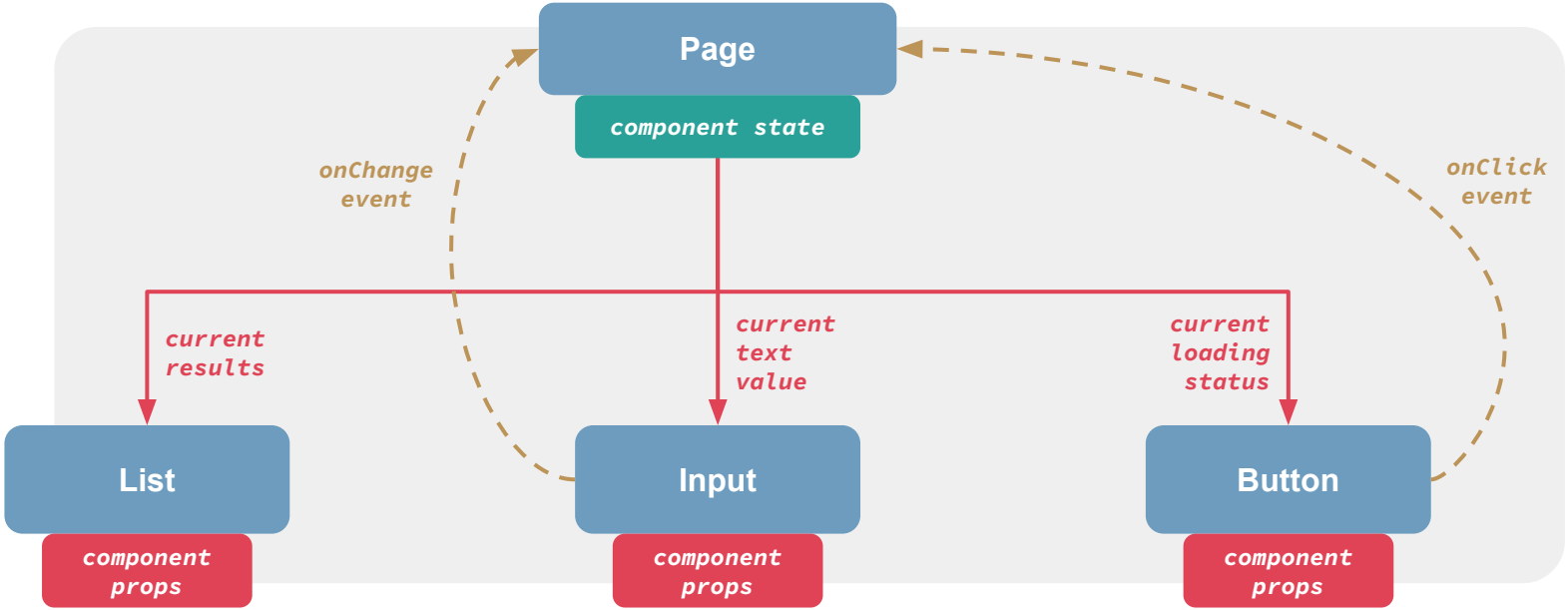

```
<WikiSearch
  title="Search here"
/>;
```

Search here

```
import React, { Component } from "react";

export class WikiSearch extends Component {
  constructor(props) {
    super(props);
    this.state = { search: [], query: null };
  }

  render() {
    const { title } = this.props;
    const { search, query } = this.state;
    return (
      <div>
        <h1>{ title }</h1>
      </div>
    );
  }
}
```

Props and state

It's all state and props



It's all state and props

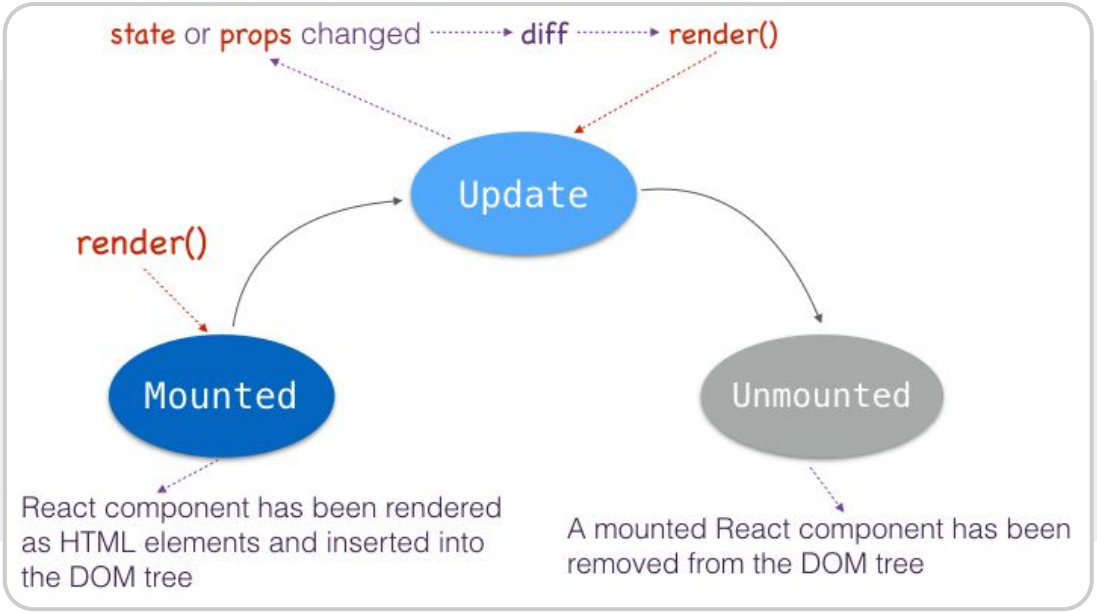

```
<div>
  <h1>{ this.props.title }</h1>

  <input
    onChange={(e) => this.setState({ query: e.currentTarget.value })}
    value={this.state.query}
  />

  <button onClick={() => this.searchWiki(this.state.query)}>
    Search
  </button>

  {this.state.search.map((s, i) => (
    <div key={i}>
      <a href={`...`}>{ s.title }</a>
      <p>{ s.snippet }</p>
    </div>
  ))}
</div>
```

It's all state and props


```
async searchWiki(value) {
  const url = `https://it.wikipedia.org/w/api.php?
               action=query&format=json&list=search&srsearch=${value}`;

  const res = await fetch(url); // call API

  if(res.ok) {
    const data = await res.json(); // get response in JSON format
    const { search } = data.query;

    this.setState({ search }); // save results into state
  } else {
    this.setState({ search: []}); // reset state
  }
}
```

Updating the state

It's all state and props



Some lifecycle steps:

→**componentDidMount**
When a component is created

→**componentDidUpdate**
When state or props are updated

→**componentWillUnmount**
When a component is destroyed

What you render is what you get

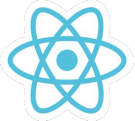
```
// our functional component
const Welcome = ({ name }) => <h1>Hello, {name}</h1>;

// the virtual DOM element
const element = <Welcome name="Dave" />;

// rendering into the web page
ReactDOM.render( element, document.getElementById("root") );
```

```
<html>
  <body>
    <div id="root"></div>
  </body>
</html>
```

```
<html>
  <body>
    <div id="root"><h1>Hello, Dave</h1></div>
  </body>
</html>
```

Render the virtual DOM to into the real one

React App Internal organization



```
package.json                // Package definition
README.md                   // Project readme
public/
  index.html                // App main HTML page
src/
  index.jsx                 // React App main entrypoint
  index.scss                // General App styles
  App.jsx                   // React App main component
  components/
    Button/                 // Component folder
       Button.jsx
       Button.scss
  pages/
    PageOne/                // Page container folder
      PageOne.pages.jsx
      PageOne.scss
  api/                      // API definitions
    users.api.js
  libs/                     // Additional libraries
    utils.js
```

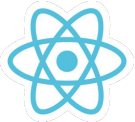
React doesn't force you to use any kind of file structure. This is a proposed approach to bring order to the various elements composing a single app.

Add styles

```
import React, { Component } from 'react';

import './App.css'; // css file styles

export default class App extends Component {
  constructor(props) {
    super(props);
  }
  render() {
    return (
      <div
        className="my-div-style"
        style={ { color: '#888', marginTop: 20, textAlign: 'center' } }
      > // inline styles </div>
    );
  }
}
```

Variation: add SCSS in Create React App

Split in components



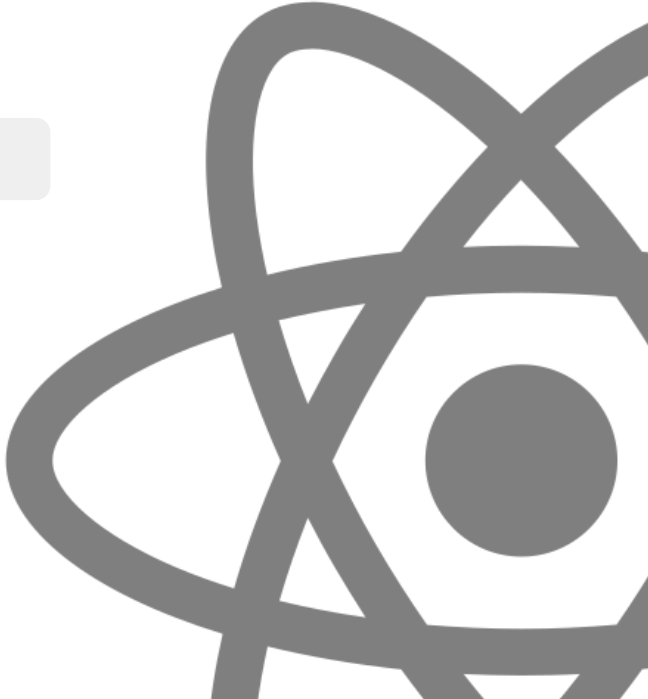
./Counter/Counter.jsx

```
import React, { Component } from 'react';

import './Counter.css'; // css file styles

export default class Counter extends Component {
  ...
}
```

./App.jsx

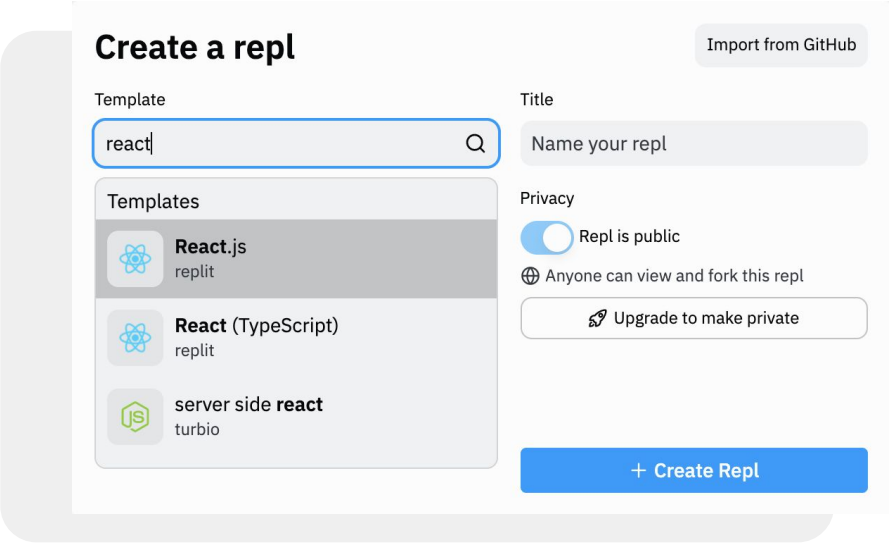
```
import Counter from ./Counter/Counter; // import custom component

export default class App extends Component {
  render() {
    return <Counter />; // include custom component
  }
}
```

Exercises



Exercises

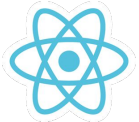
Online IDE --> https://replit.com/



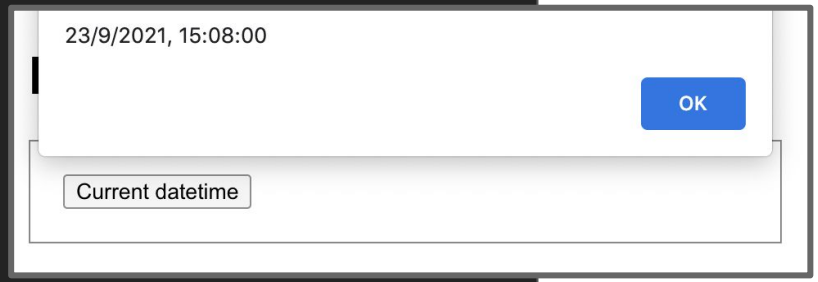
Exercises - Datetime alert

Variation: display time in page with state

Exercises - Datetime alert

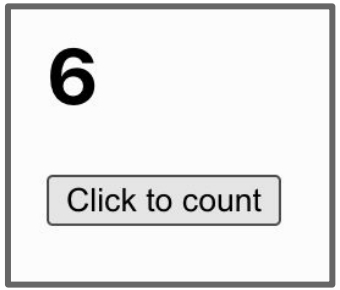


```
import React, { Component } from 'react';

export default class App extends Component {
  constructor(props) {
    super(props);
  }
  render() {
    return (
      <div>
        // create here a button to open an alert (() => alert(????))
        // Datetime string: (new Date()).toLocaleString()
      </div>
    );
  }
}
```



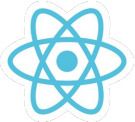
Variation: display time in page with state

Exercises - Counter



Variation: random dice throw

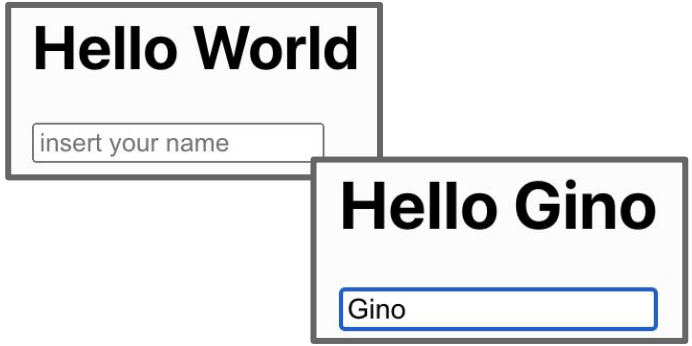
Exercises - Counter



```
import React, { Component } from 'react';

export default class App extends Component {
  constructor(props) {
    super(props);
    this.state = { // state initialization }
  }
  render() {
    const { ???? } = this.state; // state retrieval
    return (
      <div>
        <h1>{ ???? }</h1> // place here the state to be shown
        // create here a button to change the state (this.setState({ ??? }))
      </div>
    );
  }
}
```

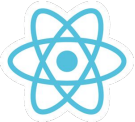
Variation: random dice throw

Exercises - Hello World



Variation: add button to confirm input

Exercises - Hello World



```
import React, { Component } from 'react';

export default class App extends Component {
  constructor(props) {
    super(props);
    this.state = { // state initialization }
  }
  render() {
    const { ???? } = this.state; // state retrieval
    return (
      <div>
        <h1>Hello { ???? }</h1> // place here the state to be shown
        // create here an input to change the state (this.setState({ ??? }))
      </div>
    );
  }
}
```

Variation: add button to confirm input

Set up a development environment

Visual Studio Code: open source IDE https://code.visualstudio.com/

npm: package manager https://www.npmjs.com/get-npm

NodeJS: runner https://nodejs.org/

git: version control system https://git-scm.com/

Create React App: official boilerplate https://create-react-app.dev/docs/getting-started/