

React Components

A FRONTEND LIBRARY

Using bootstrap in react app

>npm i bootstrap

>npm install font-awesome

Then you can use

Use below code in either index.js or in any component

```
import "bootstrap/dist/css/bootstrap.css";
```

```
import "font-awesome/css/font-awesome.css";
```

Function and Class Components

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

Using a Component

```
<Welcome name="Sara" />
```

Attributes passed to a component are props

Props are read only

Don't be afraid to split components into smaller components.

HTML ? FancyBorder Tag?

```
<FancyBorder color="blue">  
    <Welcome name="Sara" />  
</FancyBorder>
```

Composing Components

```
function App() {  
  return (  
    <div>  
      <Welcome name="Sara" />  
      <Welcome name="Cahal" />  
      <Welcome name="Edite" />  
    </div>  
  );  
}
```

Welcome Component

```
import React from "react";
const Welcome= (props) => {
    return <div>Welcome {props.name}!</div>;
};
export default Welcome;
```

FancyBorder

```
import React from "react";
const FancyBorder = (props) => {
  return (
    <div style={{ border: "1px solid", borderColor: props.color }}>
      {props.children}
    </div>
  );
};
export default FancyBorder;
```

Welcome Component inside FancyBorder

```
<FancyBorder color="blue">  
    <Welcome name="Sara" />  
</FancyBorder>
```

Using a Component

```
import Welcome `from  
"./components/examples/Welcome";  
  
ReactDOM.render(<Welcome name="Usman"/>,  
document.getElementById("root"));
```

//instead of using App we are using Welcome

Using a Component inside another

```
import React from "react";
import Welcome from "./Welcome";
const ManyWelcome = () => {
  return (
    <div>
      <Welcome name="Usman" />
      <Welcome name="Hassan" />
    </div>
  );
};
export default ManyWelcome;
```

Component can return only a Single Element

```
import React from "react";
const Welcome= (props) => {
  return <div>Welcome {props.name}!</div>;
};
export default Welcome;
```

Embedding Expressions

```
const count=0;  
  return (  
    <div className="App">  
      {count}  
    </div>  
  );
```

If Short Hand

```
{count === 0 ? "Zero" : count}
```

```
{count !== 0 && count} //print count if not zero
```

State

```
const [count, setCount] = React.useState(5);  
//A special variable to control Component  
State
```

React will remember its current value between re-renders,
Provide the most recent one to our function.
Update the current count, we can call setCount.

Counter With State

```
{count < 5 && <div>Count is in dangerous state</div>}
```

```
{count < 5 ? <div>Red Counter</div> : <div>Green Counter</div>}
```

Object Destructuring

```
var o = {p: 42, q: true};
```

```
var {p, q} = o;
```

```
console.log(p); // 42
```

```
console.log(q); // true
```

A Clean Approach

```
formatCount() {  
    const count=7;  
    return count === 0 ? "Zero" : count;  
}  
  
return (  
    <div>  
        <span>{formatCount()}</span>  
    </div>  
);
```

formatCount can also return jsx expression

```
formatCount() {  
    const count=5;  
    return count === 0 ? "Zero" :  
        <span> {count}</span>;  
}
```

Setting Attributes

```
return <img src={imageUrl} alt="" />;
```

Setting CSS Classes

```
const count = 0;  
return (  
  <div className="App">  
    {count === 0 ? "Zero" : count}  
    {count !== 0 && count}  
  </div>  
);
```

Setting Styles from jsx

```
styles = {  
  fontWeight: 'bold'  
}  
  
const count = 0;  
  
return (  
  <div className="App" style={styles}>  
    Hello  
  </div>  
);
```

Dynamic Classes

```
getBadgeClasses() {  
  let classes = "badge m-2 badge-";  
  classes += count === 0 ? "warning" : "primary";  
  return classes;  
}  
  
//Then in component  
//className={getBadgeClasses()}
```

Handling Lists

```
consttags: ['Funny', 'SciFi', 'Horror'];  
return (  
  <ul>  
    {tags.map(tag => <li>{tag}</li>)})  
  </ul>  
);  
}  
//This will generate Warning. Each li should have a  
key
```

Specifying Key

```
const numbers = [1, 2, 3, 4, 5];  
const listItems = numbers.map((number) =>  
  <li key={number.toString()}>  
    {number}  
  </li>  
);
```

Specifying Key

```
const todoltems = todos.map((todo, index) =>
  // Only do this if items have no stable IDs
  <li key={index}>
    {todo.text}
  </li>
);
```

Conditional Rendering

```
renderTags() {  
    if(tags.length === 0) return <p>No Tags</p>;  
    return tags.map(tag => <li>{tag}</li>);  
}  
  
//Then in Render  
//{tags.length === 0 && "Please Create Tags"}  
//{renderTags()}
```

Handling Events

Raising an Event

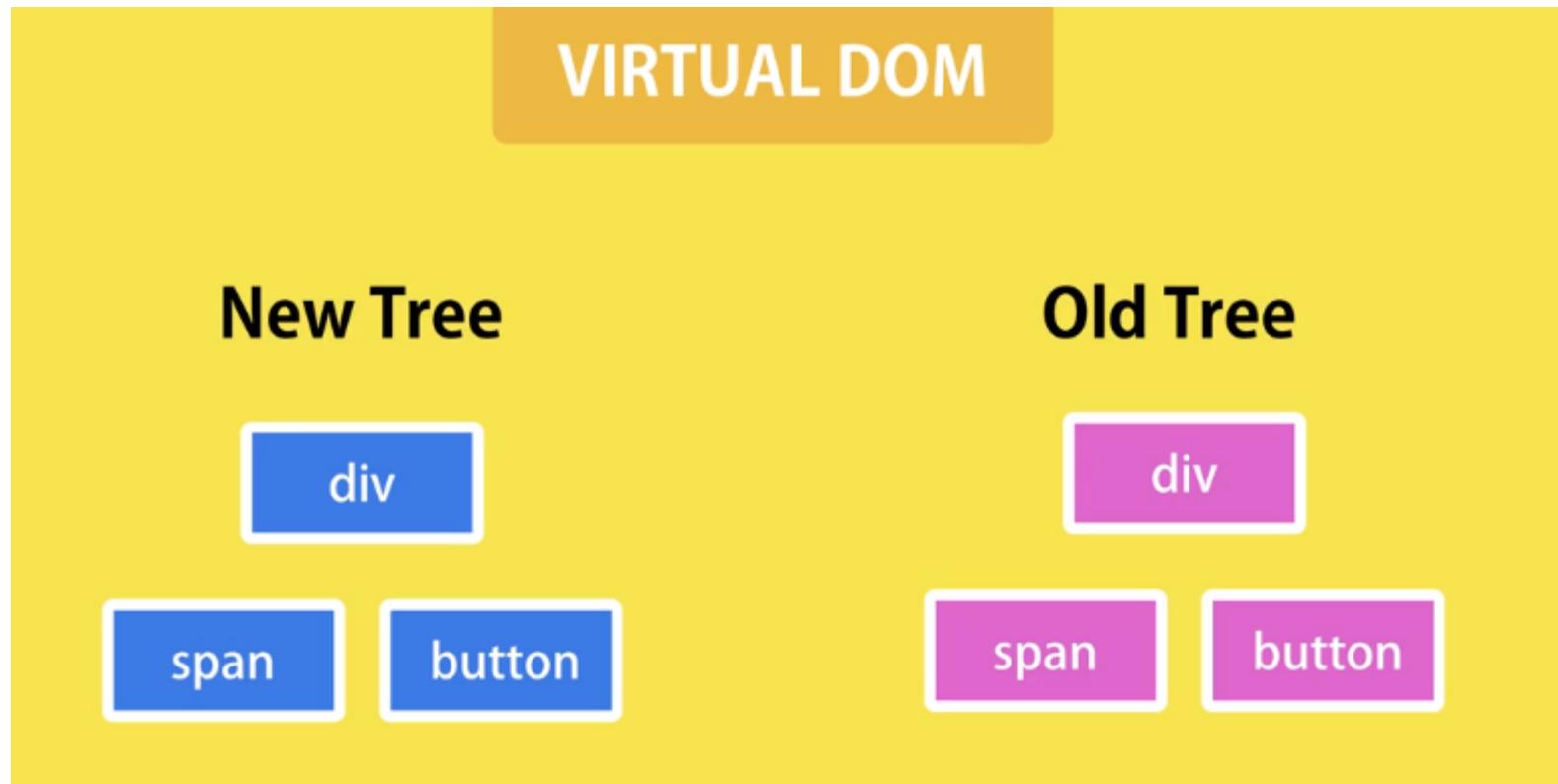
```
<Button variant="contained" color="primary" onClick={countUp}>  
  <AddIcon />  
</Button>
```

Handling Events

Capturing an Event

```
const countUp = () => {  
    setCount(count + 1);  
    // count++;  
    // alert(count);  
};
```

What Happens when state change



Passing Parameters to event handler

```
handleIncrement = (id) => {  
  //Handle ID  
};
```

Call it like this

```
onClick={()=>handleIncrement(3)}
```

Vidly Project

<https://1drv.ms/f/s!AtGKdbMmNBGd1GXjgiJoFilmccTx>

>npm install

>npm start

Showing 9 movies in the database.

Title	Genre	Stock	Rate	
Terminator	Action	6	2.5	<button>Delete</button>
Die Hard	Action	5	2.5	<button>Delete</button>
Get Out	Thriller	8	3.5	<button>Delete</button>
Trip to Italy	Comedy	7	3.5	<button>Delete</button>
Airplane	Comedy	7	3.5	<button>Delete</button>
Wedding Crashers	Comedy	7	3.5	<button>Delete</button>
Gone Girl	Thriller	7	4.5	<button>Delete</button>
The Sixth Sense	Thriller	4	3.5	<button>Delete</button>
The Avengers	Action	7	3.5	<button>Delete</button>

Composing Components

<https://1drv.ms/f/s!AtGKdbMmNBGd1zi0xRJmY1kA7tqt>

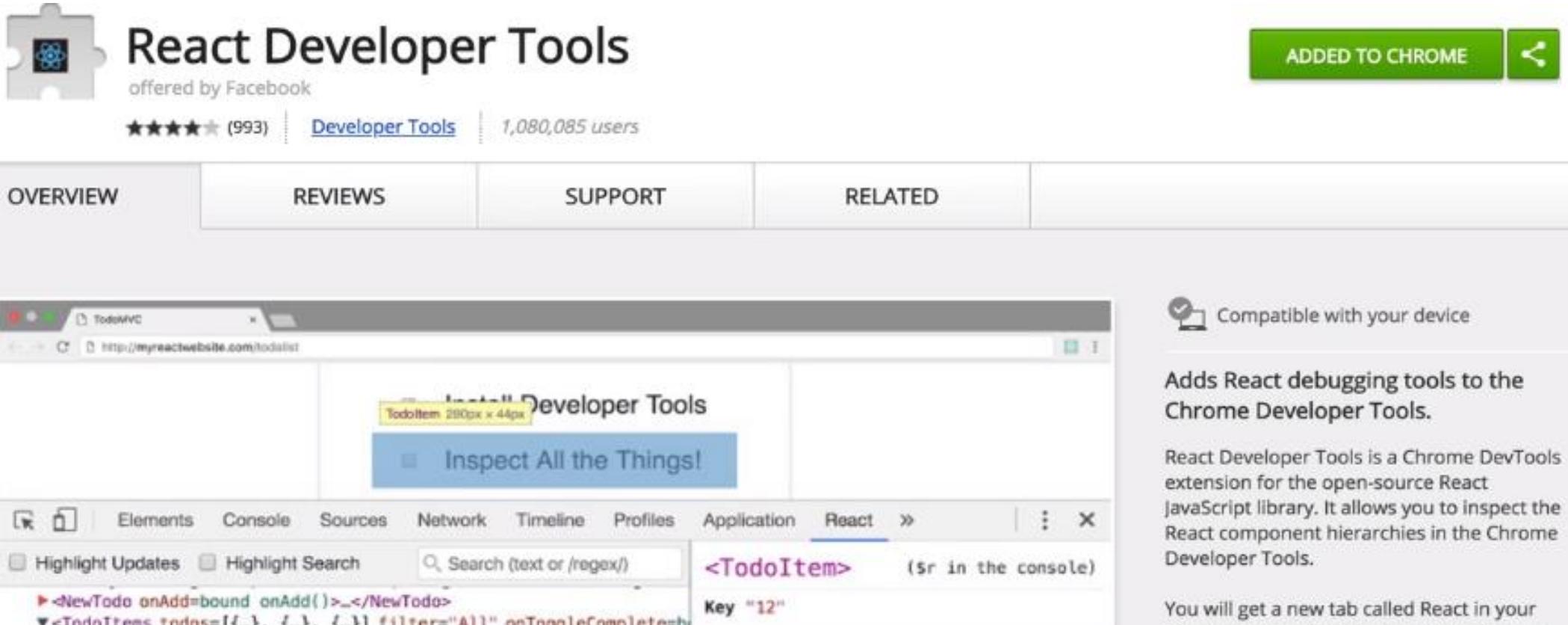
Start to Finish

Navbar 2

The image shows a user interface composed of three identical components, each with a number (3, 2, or Zero) in a yellow box, followed by a dark gray plus button, a dark gray minus button, and a red Delete button.

Value	+	-	Delete
3	+	-	Delete
2	+	-	Delete
Zero	+	-	Delete

React Debugging



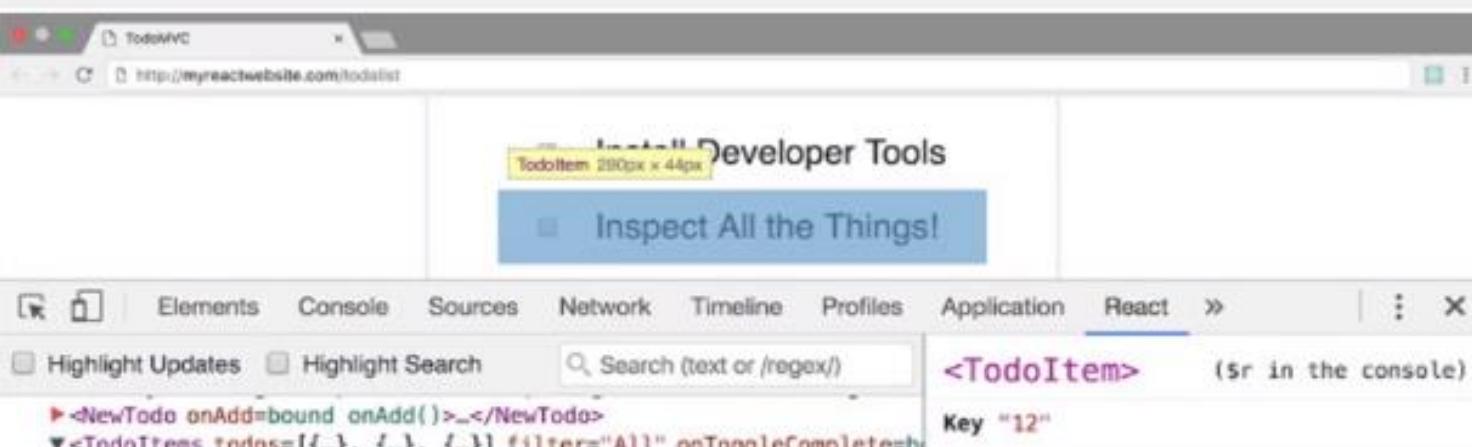
The image shows the 'React Developer Tools' extension page on the Chrome Web Store. The page includes a logo, developer information, user reviews, and a 'REVIEW' button. Below the review section is a screenshot of the Chrome DevTools interface with the React tab selected, showing component hierarchy and state. To the right of the screenshot, there's descriptive text about the extension's compatibility and its purpose of adding React debugging tools to the Chrome Developer Tools.

React Developer Tools
offered by Facebook

★★★★★ (993) | [Developer Tools](#) | 1,080,085 users

[REVIEW](#)

OVERVIEW **REVIEWS** **SUPPORT** **RELATED**

A screenshot of the Chrome DevTools interface with the React tab selected. The React tab shows a component hierarchy with '<TodoItem>' at the top level and 'Key "12"' below it. The interface also includes tabs for Elements, Console, Sources, Network, Timeline, Profiles, Application, and a search bar.

Compatible with your device

Adds React debugging tools to the Chrome Developer Tools.

React Developer Tools is a Chrome DevTools extension for the open-source React JavaScript library. It allows you to inspect the React component hierarchies in the Chrome Developer Tools.

You will get a new tab called React in your browser.

Single Source of Truth

Try to make Stateless components as much as possible.

Data should be kept at only one location

E.g. Movies Components has many SingleMovie Components then movies data should only be placed in parent .

Passing Functions as props

```
<Like  
    liked={movie.liked}  
    onClick={() => handleLike(movie)}  
/>
```

Calling Passed Function (Like.jsx)

```
<i  
  onClick={props.onClick}  
  style={{ cursor: "pointer" }}  
  className={classes}  
  aria-hidden="true"  
/>
```