

LECTURE #1

By: Aqib Rehman

OUTLINE

React vs React Native

React Intro

How React works

React setup

React Components

State

React Native



Use React to create mobile apps (Android & iOS)



Great choice for developers already familiar with React for web

REACT

React: A Front-end Development Library

React, or ReactJs is an open-source JavaScript library used to develop single-page web applications. It's one of the popular libraries to build UI (front-end) for web and mobile apps. It was developed and backed by Facebook in 2011 and has gained popularity since then.



React uses the Virtual DOM to create amazing UX, while React Native leverages APIs to render UI components that can be used again for both Android and iOS platforms.

REACT NATIVE ALSO USED:

Core components of React

i.e. Functional Components, States, Props

React Hooks and Context

JavaScript Basic Fundamentals

i.e. objects, arrays, error functions, callbacks

Without JavaScript knowledge not a good place to start

REACT

What is React?

- JavaScript library created by Facebook**
- Also used by Netflix & Instagram**
- Used to create JS-driven dynamic web apps**
- In that regard, can be compared to Angular & Vue**

What is Redux?

- A layer on-top of React
- Helps with state management of our app
 - data in the app
 - UI state of the app

State management means:

Is pop up open or closed

Is menu open or closed

- React basics
 - components, events, templates, props & forms
- React Router
 - routes, route parameters, redirects
- Redux
 - stores, actions & reducers

React

- Components, components, COMPONENTS!



The Virtual DOM

- The Virtual DOM makes React fast...



Components & Templates

- Components look like HTML templates (actually JSX)
- They can contain 'state' (data or UI state)
- They also can contain JavaScript for functionality

```
class App extends React.Component {  
  state = {  
    name: "Ryu",  
    age: 30  
  }  
  render(){  
    return(  
      <div className="app-content">  
        <h1>Hello, ninjas!</h1>  
        <p>My name is: { this.state.name } and I am { this.state.age }</p>  
      </div>  
    )  
  }  
}
```

index.html - react-redux-complete - Visual Studio Code

EXPLORER index.html

- OPEN EDITORS 1 UNSAVED
- index.html

- REACT-REDUX-COMPLETE
- index.html

```
1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <title>Document</title>
7 </head>
8 <body>
9
10 </body>
11 </html>
```

A screenshot of the Visual Studio Code interface. The title bar shows "index.html - react-redux-complete - Visual Studio Code". The left sidebar has icons for Explorer, Open Editors (1 unsaved), and React-Redux-Complete, with "index.html" selected under the latter. The main editor area shows the following code:

```
1 <html>
2   <head>
3     <meta charset="UTF-8">
4     <meta name="viewport" content="width=device-width, initial-scale=1.0">
5     <meta http-equiv="X-UA-Compatible" content="ie=edge">
6     <script src="https://unpkg.com/react@16/umd/react.development.js"></script>
7     <script src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8   </head>
9   <body>
10  <h1>Hello from React</h1>
11  <p>This is a simple React app.</p>
12  <button>Click me</button>
13</body>
```

```
<script crossorigin  
src="https://unpkg.com/react@18/umd/react.development.js"></script>  
<script crossorigin src="https://unpkg.com/react-dom@18/umd/react-  
dom.development.js"></script>
```

+ VS Code editor - <https://code.visualstudio.com/>

index.html - react-redux-complete - Visual Studio Code

EXPLORER

OPEN EDITORS 1 UNSAVED

- index.html

REACT-REDUX-COMPLETE

- index.html

```
1 <html lang="en">
2   <head>
3     <meta charset="UTF-8">
4     <meta name="viewport" content="width=device-width, initial-scale=1.0">
5     <meta http-equiv="X-UA-Compatible" content="ie=edge">
6     <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
7     <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8     <title>Document</title>
9   </head>
10  <body>
11
12    <div id="app"></div>
13
14  </body>
15 </html>
```

PACKAGES IN VS-CODE

Extensions: ES7 React/Redux/GraphQL/React-Native snippets - react-redux-complete - Visual Studio Code

EXTENSIONS

Search Extensions in Marketplace

ENABLED

- ES7 React/Redux/GraphQL/React-Native snippets 1.8.5 dsnajder | 441,810 installs | ★★★★★ | Repository
- Simple extensions for React, Redux, GraphQL and React Native
- Live Server 5.1.1 Ritwick Dey
- Material Icon Theme 3.3.2 Philipp Kieff
- Monokai++ 1.6.7 Davide Casella
- Sublime Babel 0.2.10 Josh Peng
- Vetur 0.12.5 Pine Wu

RECOMMENDED

- TSLint 1.0.34 TSLint for Visual Studio Code

index.html Extension: ES7 React/Redux/GraphQL/React-Native snippets

ES7 React/Redux/GraphQL/React-Native snippets dsznajder/es7-react-js-snippets

dsnajder | 441,810 | ★★★★★ | Repository

Simple extensions for React, Redux and GraphQL in JS/TS with ES7 syntax

[Disable](#) [Uninstall](#)

[Details](#) [Contributions](#) [Changelog](#) [Dependencies](#)

VS Code ES7 React/Redux/React-Native/JS snippets

Visual Studio Marketplace v1.8.5 installs 441,810 rating 5/5 (7)

This extension provides you Javascript and React/Redux snippets in ES7 with babel plugins features for VS Code.

Here is direct link to marketplace [ES7 React/Redux/React-Native/JS Snippets](#)

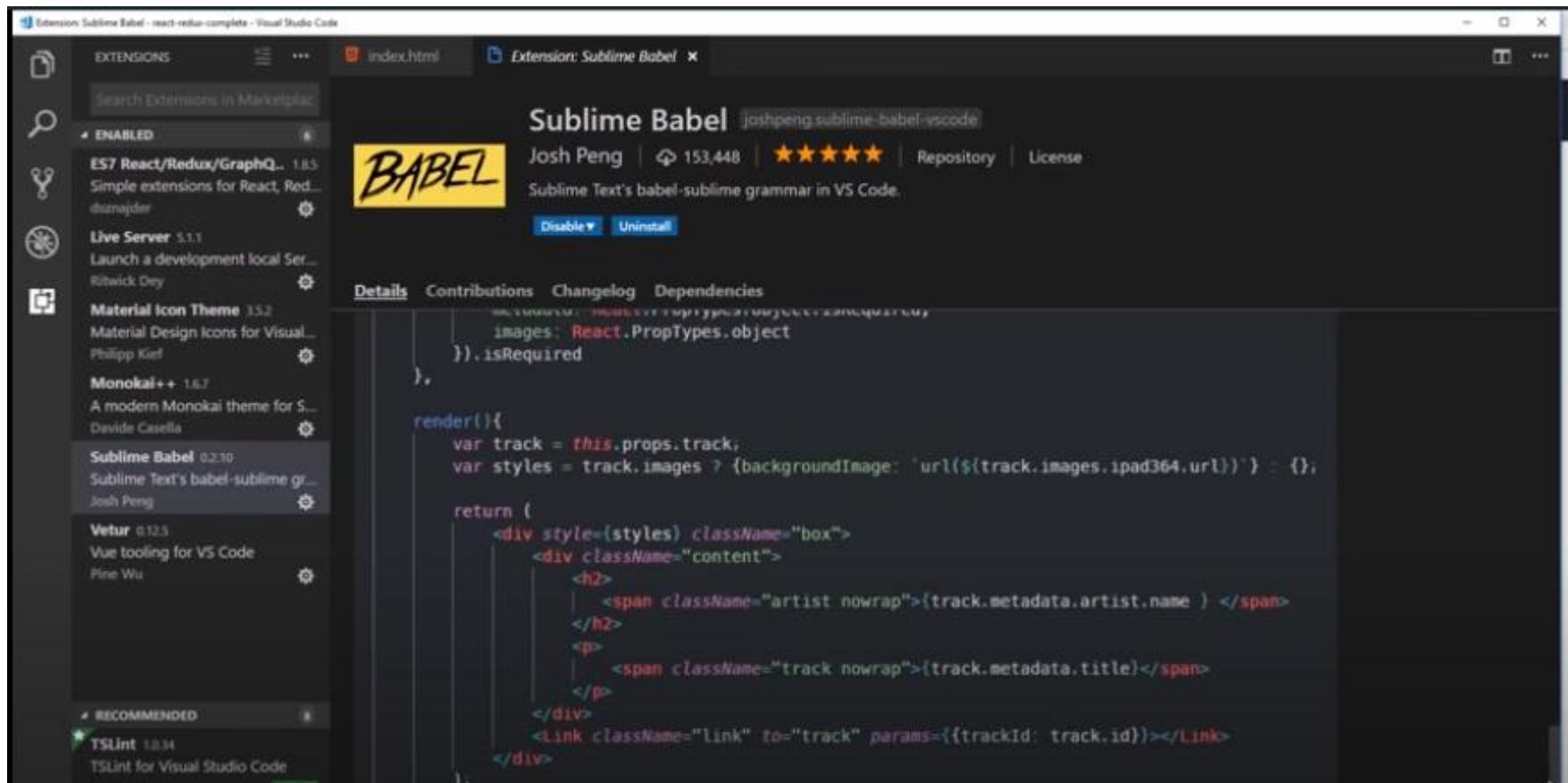
Supported languages (file extensions)

- JavaScript (.js)
- JavaScript React (.jsx)
- TypeScript (.ts)
- TypeScript React (.tsx)

Snippets info

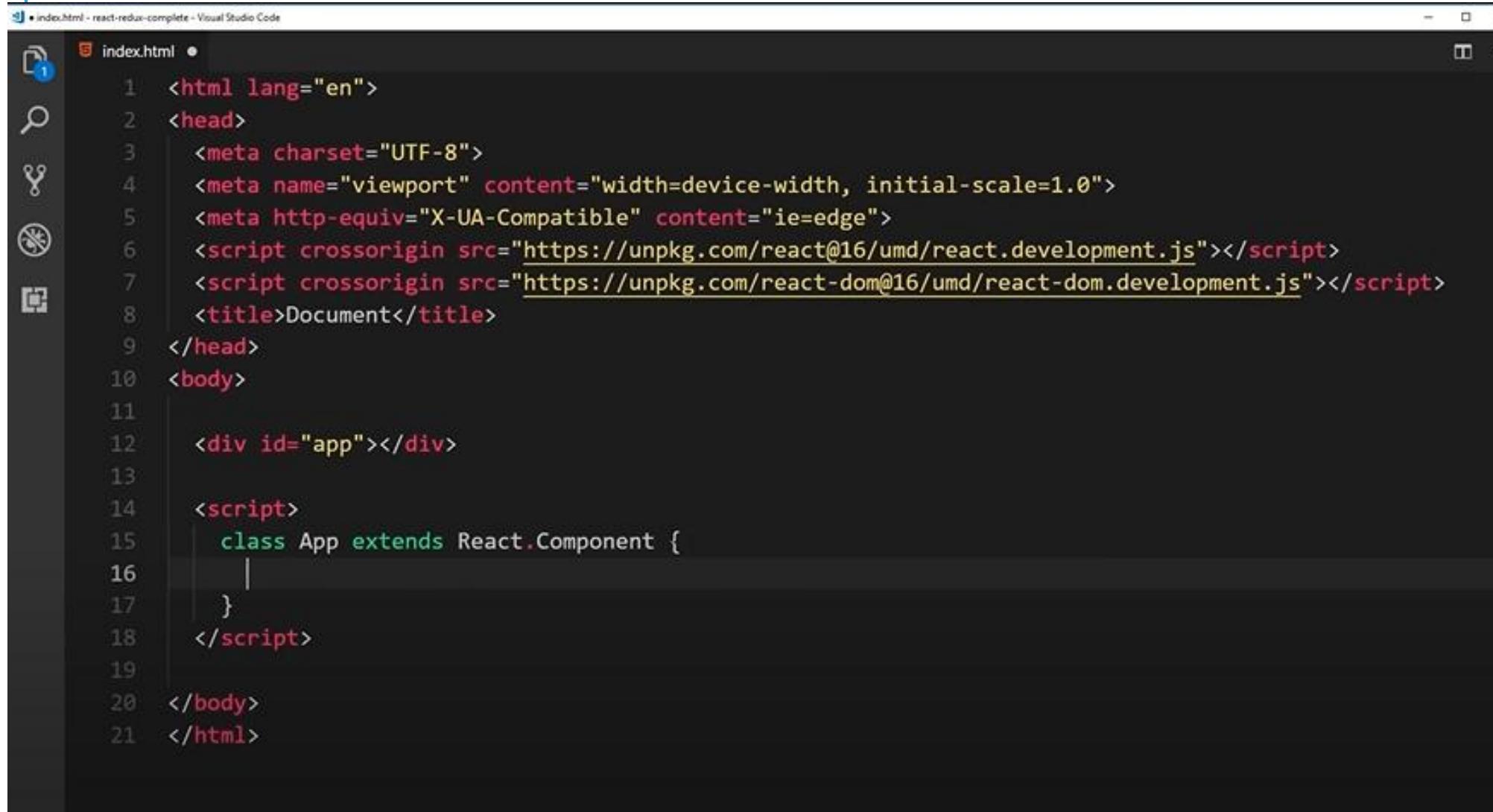
Every space inside { ... } and (...) means that this is pushed into next line.) \$ represent each step after tab.

The screenshot shows the Visual Studio Code interface with the Extensions sidebar open. The 'Enabled' section lists several extensions, with 'ES7 React/Redux/GraphQL/React-Native snippets' by dsnajder being the top item. This extension has 441,810 installs and a perfect 5/5 rating. The details page for this extension is displayed, featuring its name, developer, stats, and a brief description. It also lists supported languages (JavaScript, JavaScript React, TypeScript, TypeScript React) and provides information about snippets, mentioning spaces and tabs. The 'RECOMMENDED' section at the bottom includes 'TSLint'.



REACT COMPONENT

CLASS BASED COMPONENT



A screenshot of the Visual Studio Code interface showing the `index.html` file. The code defines a basic HTML structure with a `head` section containing meta tags for charset, viewport, and compatibility, and a `body` section with a `div` element of `id="app"`. Inside the `body`, there is a `script` tag that defines a class-based React component named `App` which extends `React.Component`. The code uses syntax highlighting to distinguish between HTML tags, CSS-like classes and IDs, and the JavaScript code within the `script` tag.

```
index.html
1 <html lang="en">
2   <head>
3     <meta charset="UTF-8">
4     <meta name="viewport" content="width=device-width, initial-scale=1.0">
5     <meta http-equiv="X-UA-Compatible" content="ie=edge">
6     <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
7     <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8     <title>Document</title>
9   </head>
10  <body>
11
12    <div id="app"></div>
13
14    <script>
15      class App extends React.Component {
16        |
17      }
18    </script>
19
20  </body>
21 </html>
```

React.com
ponent=
React will
be used
from React
library
and
React.Com
ponent will
get all the
basics of
component

In 2018 conference hooks introduced.

React 16.8.0 is the first release to support Hooks (launch in 2019).

In 2023 new documentation website launch for React 18 and onwards.

<https://legacy.reactjs.org/docs/hooks-intro.html>

<https://react.dev/versions>

There are no plans to remove classes from React.

<https://legacy.reactjs.org/docs/hooks-intro.html>

A class based component must have atleast 1 function that is render



index.html

```
1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
7   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8   <title>Document</title>
9 </head>
10 <body>
11
12   <div id="app"></div>
13
14   <script>
15     class App extends React.Component {
16       render() {}
17     }
18   </script>
19
20 </body>
21 </html>
```



index.html - react-redux-complete - Visual Studio Code

index.html

```
1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
7   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8   <title>Document</title>
9 </head>
10 <body>
11
12   <div id="app"></div>
13
14   <script>
15     class App extends React.Component {
16       render() {
17         return (
18           <div>
19             <h1>Hey, ninjas</h1>
20           </div>
21         )
22       }
23     }
24   </script>
25 
```

In jsx only one root element can be defined

CAN'T USED LIKE THIS

The screenshot shows a Visual Studio Code window with the file "index.html" open. The code is as follows:

```
1 <html lang="en">
2   <head>
3     <meta charset="UTF-8">
4     <meta name="viewport" content="width=device-width, initial-scale=1.0">
5     <meta http-equiv="X-UA-Compatible" content="ie=edge">
6     <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
7     <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8     <title>Document</title>
9   </head>
10  <body>
11
12    <div id="app"></div>
13
14    <script>
15      class App extends React.Component {
16        render() {
17          return (
18            <div>
19              <h1>Hey, ninjas</h1>
20            </div>
21            <h1>
22              <h3>
23            )
24          }
25        }
      
```

A red rectangular box highlights the code from line 21 to line 25, specifically the nested `<h1>` and `<h3>` tags. The Babel extension icon is visible in the status bar.

Instead of class in javascript we will use classname in jsx

A screenshot of the Visual Studio Code interface, showing the file `index.html` open. The code is a simple HTML document with a React component. The code editor has a dark theme with syntax highlighting. The sidebar on the left contains icons for file operations like new file, save, and search.

```
1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
7   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8   <title>Document</title>
9 </head>
10 <body>
11
12   <div id="app"></div>
13
14   <script>
15     class App extends React.Component {
16       render() {
17         return (
18           <div className="app-content">
19             <h1>Hey, ninjas</h1>
20           </div>
21         )
22       }
23     }
24   </script>
25 
```

index.html •

```
7   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
8   <title>Document</title>
9   </head>
10  <body>
11
12  <div id="app"></div>
13
14  <script>
15    class App extends React.Component {
16      render() {
17        return (
18          <div className="app-content">
19            <h1>Hey, ninjas</h1>
20          </div>
21        )
22      }
23    }
24
25    ReactDOM.render(<App />, document.getElementById('app'))
26  </script>
27
28  </body>
29  </html>
```



EXTENSIONS



index.html x



Search Extensions in Marketplace

ENABLED

ES7 React/Redux/GraphQL 1.8.5
Simple extensions for React, Red...
dhananjdeLive Server 5.1.1
Launch a development local Ser...
Ritwick Dey

Material Icon Theme 1.5.2

Material Design Icons for Visual...
Philipp KiefMonokai++ 1.6.7
A modern Monokai theme for S...
David CasellaSublime Babel 0.2.10
Sublime Text's babel-sublime gr...
Josh PengVetur 0.12.3
Vue tooling for VS Code
Pine Wu

RECOMMENDED

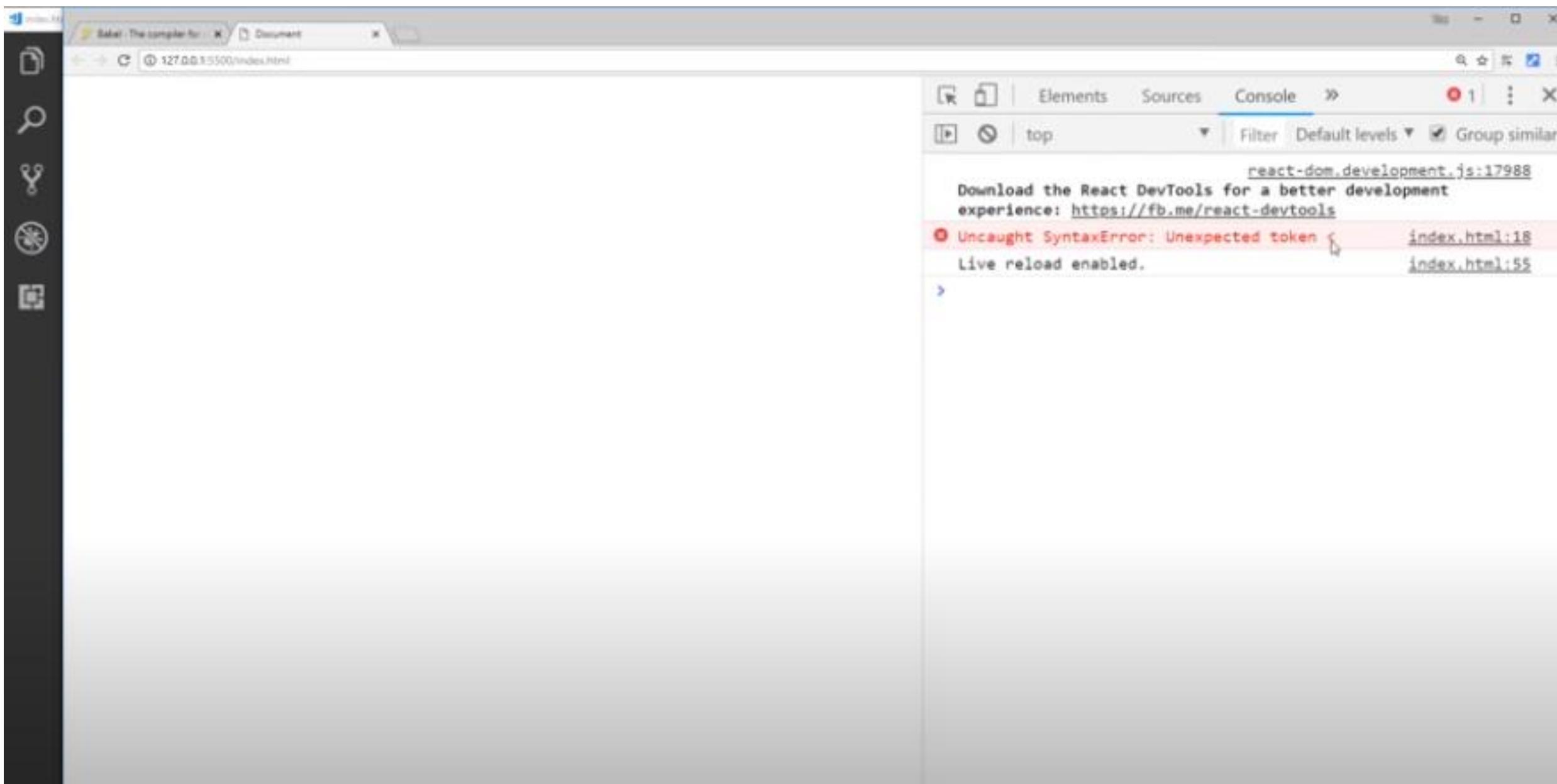
TSLint 1.0.34
TSLint for Visual Studio Code

```
7   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js">
8     <title>Document</title>
9   </head>
10  <body>
11
12    <div id="app"></div>
13
14  <script>
15    class App extends React.Component {
16      render() {
17        return (
18          <div className="app-content">
19            <h1>Hey, ninjas</h1>
20          </div>
21        )
22      }
23    }
24
25    ReactDOM.render(<App />, document.getElementById('app'));
26  </script>
27
28  </body>
```

BABEL



Right click: open in live server



The screenshot shows a browser window with developer tools open. The address bar indicates the page is at `127.0.0.1:5500/index.html`. The developer tools sidebar on the left has icons for file, search, and refresh. The main area shows the `Sources` tab selected, displaying the code for `index.html`. The code is a React component named `App`:

```
1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script crossorigin src="https://unpkg.com/react@16.13.1/dist/react.js">
7   <script crossorigin src="https://unpkg.com/react-dom@16.13.1/dist/react-dom.js">
8   <title>Document</title>
9 </head>
10 <body>
11   <div id="app"></div>
12
13   <script>
14     class App extends React.Component {
15       render() {
16         return (
17           <div className="app-content">
18             <h1>Hey, ninjas!</h1>
19           </div>
20         )
21       }
22     }
23   </script>
24
25   ReactDOM.render(<App />, document.getElementById('app'))
26 </script>
27
28 <!-- Code injected by live-server -->
29 <script type="text/javascript">
30   // <![CDATA[ <-- For SVG support
31   // <!-- This part is injected by live-server -->
```

A yellow highlight covers the line `<div className="app-content">`, and a blue highlight covers the line `<h1>Hey, ninjas!</h1>`.

Not understanding jsx: because jsx is not supported in browsers

The screenshot shows a Microsoft Edge browser window with the URL <https://babeljs.io/setup#installation>. The page is titled "BABEL 6.x" and features a navigation bar with links to "Docs", "Setup", "Try it out", "Blog", "Search", "Donate", "Team", and "GitHub". On the left, there's a vertical sidebar with icons for file operations like Open, Save, Print, and Find. The main content area displays information about Babel's own REPL, online editors (JSFiddle, JSBin, Codepen), and usage examples.

Online Editors that run Babel for you:

- [JSFiddle](#)
- [JSBin](#)
- [Codepen](#)

3 Usage

With babel-standalone

```
HTML: <div id="output"></div>
<!-- Load Babel -->
<script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
<!-- Your custom script here -->
<script type="text/babel">
const getMessage = () => "Hello World";
document.getElementById('output').innerHTML = getMessage();
</script>
```

4 Create .babelrc configuration file

Great! You've configured Babel but you haven't made it actually do anything. Create a `.babelrc` config in your project root and enable some plugins.

To start, you can use the `env` preset, which enables transforms for ES2015+

```
index.html • 1 <html lang="en">
2 <head>
3   <meta charset="UTF-8">
4   <meta name="viewport" content="width=device-width, initial-scale=1.0">
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
7   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
8   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
9   <title>Document</title>
10 </head>
11 <body>
12
13   <div id="app"></div>
14
15   <script>
16     class App extends React.Component {
17       render() {
18         return (
19           <div className="app-content">
20             <h1>Hey, ninjas</h1>
21           </div>
22         )
23     }
24   </script>
```

BABEL 6.x

Docs Setup Try it out Blog Search

Donate Team GitHub

With babel-standalone

HTML

Copy

```
<div id="output"></div>
<!-- Load Babel --&gt;
&lt;script src="https://unpkg.com/babel-standalone@6/babel.min.js"&gt;&lt;/script&gt;
<!-- Your custom script here --&gt;
&lt;script type="text/babel"&gt;
const getMessage = () =&gt; "Hello World";
document.getElementById('output').innerHTML = getMessage();
&lt;/script&gt;</pre>
```

4 Create .babelrc configuration file

Great! You've configured Babel but you haven't made it actually do anything. Create a `.babelrc` config in your project root and enable some plugins.

To start, you can use the `env` preset, which enables transforms for ES2015+

Shell

Copy

index.html - react-redux-complete - Visual Studio Code

index.html

```
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
7   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
8   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
9   <title>Document</title>
10  </head>
11  <body>
12
13  <div id="app"></div>
14
15  <script type="text/babel">
16    class App extends React.Component {
17      render() {
18        return (
19          <div className="app-content">
20            <h1>Hey, ninjas</h1>
21          </div>
22        )
23      }
24    }
25
26    ReactDOM.render(<App />, document.getElementById('app'));
27  </script>
28
```

Hey, ninjas

```
 53     else if (msg.data == 'refreshc':
 54     );
 55     if (sessionStorage && !sessionStor:
 56       console.log('Live reload enabl:
 57       sessionStorage.setItem('IsThis!
 58     }
 59   })();
 60 }
 61 else {
 62   console.error('Upgrade your browser. Ti
 63 }
 64 // ]]>
 65 </script></body>
 66 </html>
```

{ } 4 lines, 92 characters selected

The screenshot shows a browser window with developer tools open. The main content area displays the text "Hey, ninjas". The developer tools sidebar includes icons for file, copy, search, and settings. The top bar shows tabs for "Babel - The compiler for..." and "Document". The address bar indicates the page is at "127.0.0.1:5100/index.html".

The Elements tab is active, showing the DOM structure:

```
<html lang="en">
  <head>...</head>
  <body>
    <div id="app">
      <div class="app-content">
        <h1>Hey, ninjas</h1> == $0
      </div>
    </div>
    <script type="text/babel">...</script>
    <!-- Code injected by live-server -->
    <script type="text/javascript">...
  </script>
  </body>
</html>
```

The `<h1>Hey, ninjas</h1>` element is selected, highlighted with a blue background. The Styles panel on the right lists the CSS properties for this element:

```
:hover .cls +, element.style {
}
h1 {
  display: block;
  font-size: 2em;
  -webkit-margin-before: 0.67em;
  -webkit-margin-after: 0.67em;
  -webkit-margin-start: 0px;
  -webkit-margin-end: 0px;
  font-weight: bold;
}
```

The bottom status bar shows the selected elements: html body div#app div.app-content h1.

index.html - react-redux-complete - Visual Studio Code

The screenshot shows the Visual Studio Code interface with a dark theme. The left sidebar contains icons for file operations like Open, Save, Find, and Refresh. The main editor area displays an HTML file named 'index.html' with the following content:

```
5 <meta http-equiv="X-UA-Compatible" content="ie=edge">
6 <script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
7 <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
8 <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
9 <title>Document</title>
10 </head>
11 <body>
12
13 <div id="app"></div>
14
15 <script type="text/babel">
16   class App extends React.Component {
17     render() {
18       return (
19         <div className="app-content">
20           <h1>Hey, ninjas</h1>
21           <p>Math.random() * 10 </p>
22         </div>
23       )
24     }
25   }

```

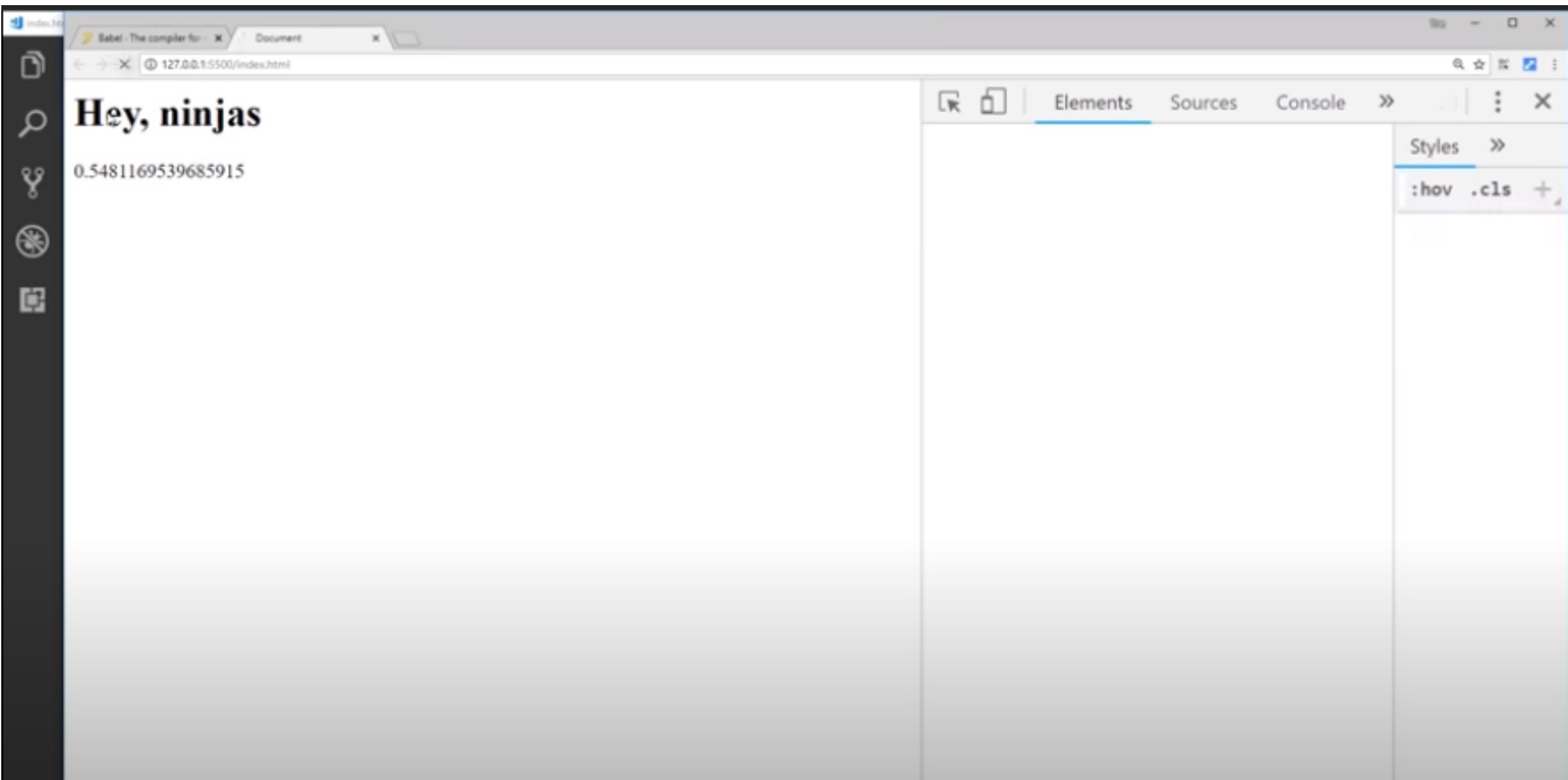
The code uses ES6 syntax, including a class-based component definition and template literals for JSX. The right side of the screen shows a vertical scroll bar and a portion of the right-hand margin where some code snippets are visible.

The screenshot shows a browser window with the address bar pointing to `127.0.0.1:5500/index.html`. The main content area displays the text "Hey, ninjas". Below the content, the browser's developer tools are open, specifically the Elements tab. A `<h1>Hey, ninjas</h1>` element is selected in the DOM tree. The right-hand panel shows the element's style definitions:

```
:hov .cls +, element.style { } user agent st... 1 { display: block; font-size: 2em; -webkit-margin-before: 0.67em; -webkit-margin-after: 0.67em; -webkit-margin-start: 0px; -webkit-margin-end: 0px; font-weight: bold; }
```



```
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
7   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
8   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
9   <title>Document</title>
10  </head>
11  <body>
12
13  <div id="app"></div>
14
15  <script type="text/babel">
16    class App extends React.Component {
17      render() {
18        return (
19          <div className="app-content">
20            <h1>Hey, ninjas</h1>
21            <p>{ Math.random() * 10 }</p>
22          </div>
23        )
24      }
25    }
26
27  ReactDOM.render(<App />, document.getElementById('app'));
```



Component State

- JavaScript Object
- Describes the current state of the component
 - data, UI-state
- The state of a component can be updated over time
 - a modal could close
 - the data we output could change

Shopping Cart Component

```
{  
  items: [  
    { name: 'navy jumper', price: 9.99 },  
    { name: 'ninja mask', price: 20.00 },  
    { name: 'black cloak', price: 15.00 }  
  ]  
}
```

State will be changed if an other item will be added in the items

Popup Component

```
{  
  showPopup: true  
}  
  
{  
  showPopup: false  
}
```

index.html - react-redux-complete - Visual Studio Code

index.html

```
5   <meta http-equiv="X-UA-Compatible" content="ie=edge">
6   <script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
7   <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
8   <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
9   <title>Document</title>
10  </head>
11  <body>
12
13  <div id="app"></div>
14
15  <script type="text/babel">
16      class App extends React.Component {
17          state = {
18
19              }
20              render() {
21                  return (
22                      <div className="app-content">
23                          <h1>Hey, ninjas</h1>
24                          <p>{ Math.random() * 10 }</p>
25                      </div>
26                  )
27              }
28      }
29  </script>
```

index.html - react-redux-complete - Visual Studio Code

index.html

```
5  <meta http-equiv="X-UA-Compatible" content="ie=edge">
6  <script src="https://unpkg.com/babel-standalone@6/babel.min.js"></script>
7  <script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script>
8  <script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>
9  <title>Document</title>
10 </head>
11 <body>
12
13 <div id="app"></div>
14
15 <script type="text/babel">
16   class App extends React.Component {
17     state = {
18       name: 'Ryu',
19       age: 30
20     }
21     render() {
22       return (
23         <div className="app-content">
24           <h1>Hey, ninjas</h1>
25           <p>{ Math.random() * 10 }</p>
26         </div>
27       )
28     }
29   }

```



index.html x

```
10  </head>
11  <body>
12
13  <div id="app"></div>
14
15  <script type="text/babel">
16      class App extends React.Component {
17          state = {
18              name: 'Ryu',
19              age: 30
20          }
21          render() {
22              return (
23                  <div className="app-content">
24                      <h1>Hey, ninjas</h1>
25                      <p>My name is: { this.state.name }</p>
26                  </div>
27              )
28          }
29      }
30
31      ReactDOM.render(<App />, document.getElementById('app'));
32  </script>
33
34  </body>
```

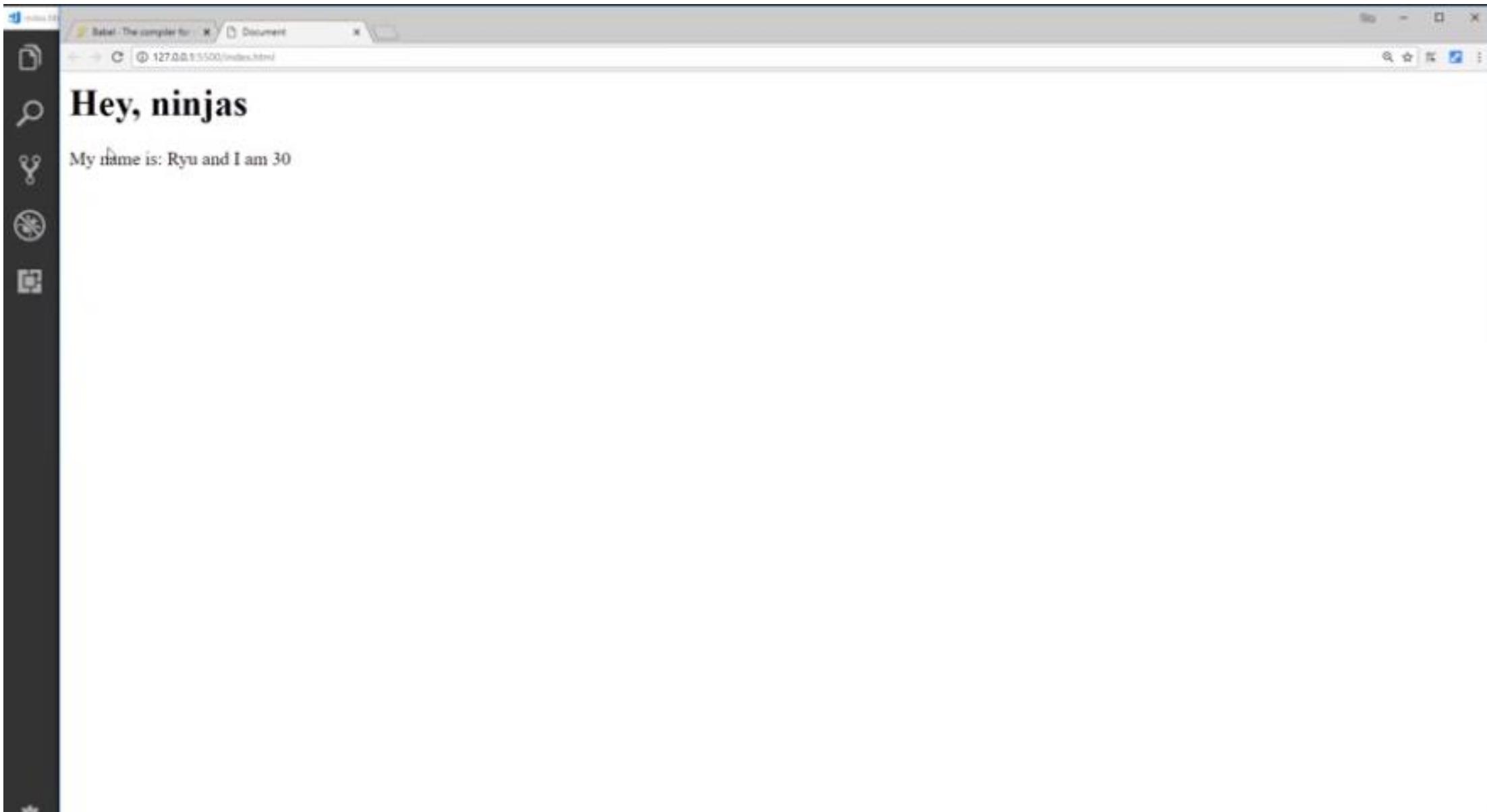


index.html - react-native-complete - Visual Studio Code

The screenshot shows a dark-themed code editor window for the file 'index.html'. The code is a simple React component definition:

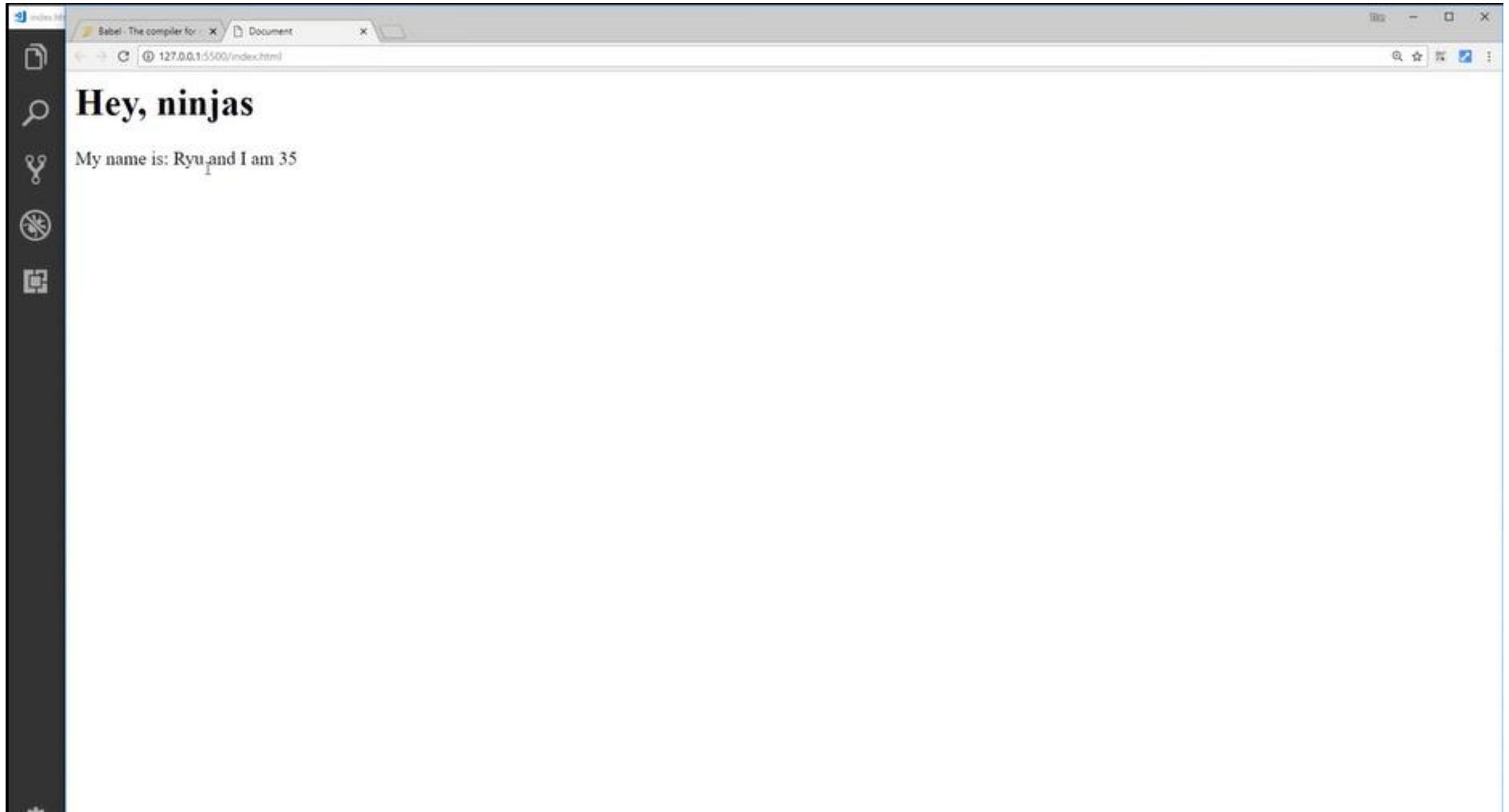
```
10  </head>
11  <body>
12
13  <div id="app"></div>
14
15  <script type="text/babel">
16      class App extends React.Component {
17          state = {
18              name: 'Ryu',
19              age: 30
20          }
21          render() {
22              return (
23                  <div className="app-content">
24                      <h1>Hey, ninjas</h1>
25                      <p>My name is: { this.state.name } and I am { this.state.age }</p>
26                  </div>
27              )
28          }
29      }
30
31      ReactDOM.render(<App />, document.getElementById('app'));
32  </script>
33
34  </body>
```

A cursor is positioned at the end of the line 'this.state.age' in the render method. A tooltip or dropdown menu is open below the cursor, containing the word 'age' preceded by a small icon. The background of the code editor has horizontal grid lines.



index.html x

```
10  </head>
11  <body>
12
13  <div id="app"></div>
14
15  <script type="text/babel">
16      class App extends React.Component {
17          state = {
18              name: 'Ryu',
19              age: 35
20          }
21          render() {
22              return (
23                  <div className="app-content">
24                      <h1>Hey, ninjas</h1>
25                      <p>My name is: { this.state.name } and I am { this.state.age }</p>
26                  </div>
27              )
28          }
29      }
30
31      ReactDOM.render(<App />, document.getElementById('app'));
32  </script>
```





Whenever a state will be changed a data will be changed in the UI