React Events

SENG 4640
Software Engineering for Web Apps
Winter 2023

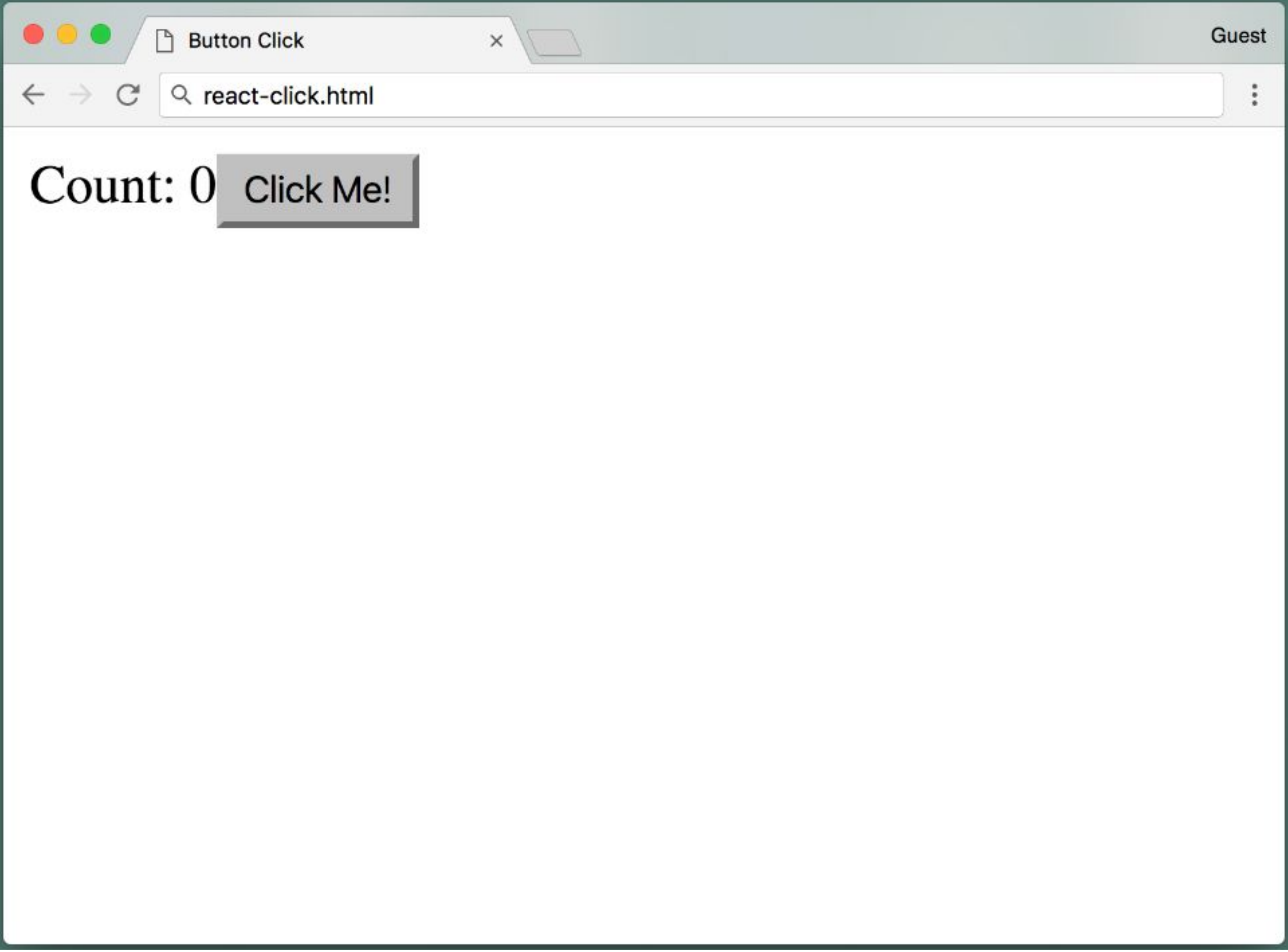
Sina Keshvadi
Thompson Rivers University

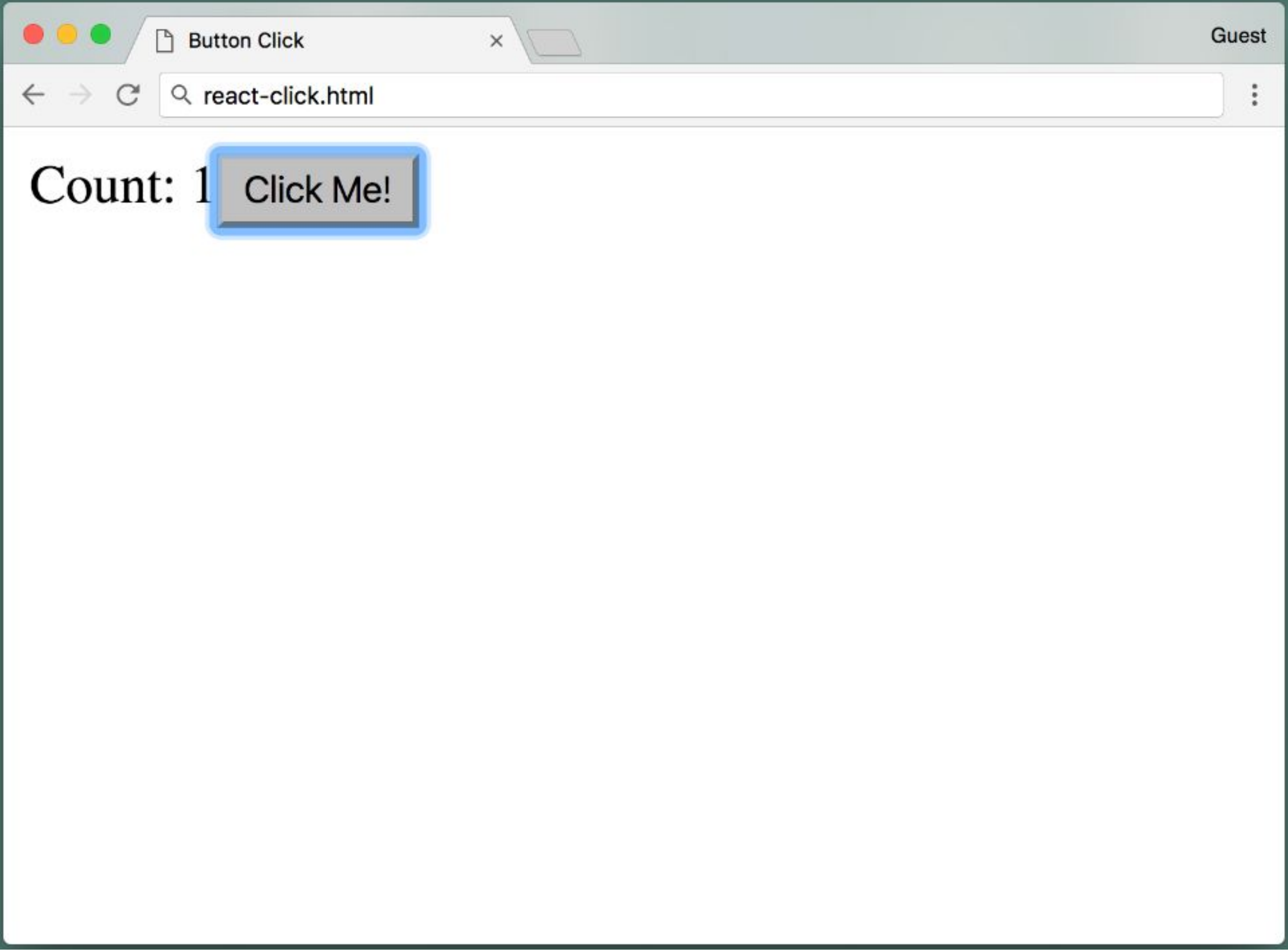
Review

- React allows us to insert JavaScript elements/components into VirtualDOM
- We can create additional components using the `React.Component` class as a base
- Components have two types of attributes
  - **Properties:** set at initialization and immutable thereafter
  - **State:** change in response to user events
- Component callback functions can be bound to HTML events

Changing Component State

- A component's state typically changes in response to some user action or "event"
- We can **bind** an event to a callback function within a React component
- That component can then change state using its **`setState`** function
- This will automatically re-render the component and any other affected component


Button Click
Guest
react-click.html
Count: 0
Click Me!



Count: 1

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

ex05.html

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

ex05.html

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

ex05.html

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

ex05.html

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

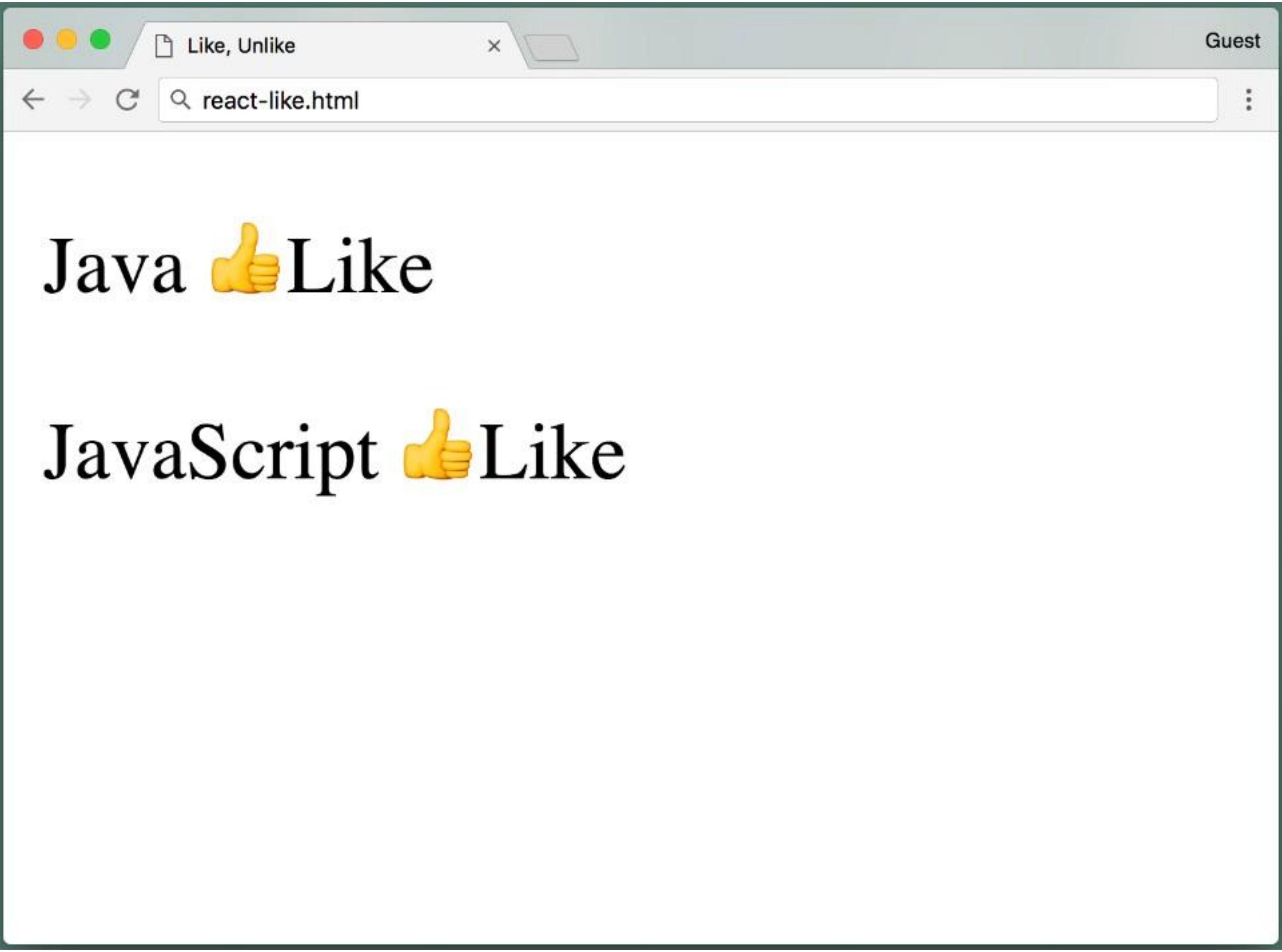
```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

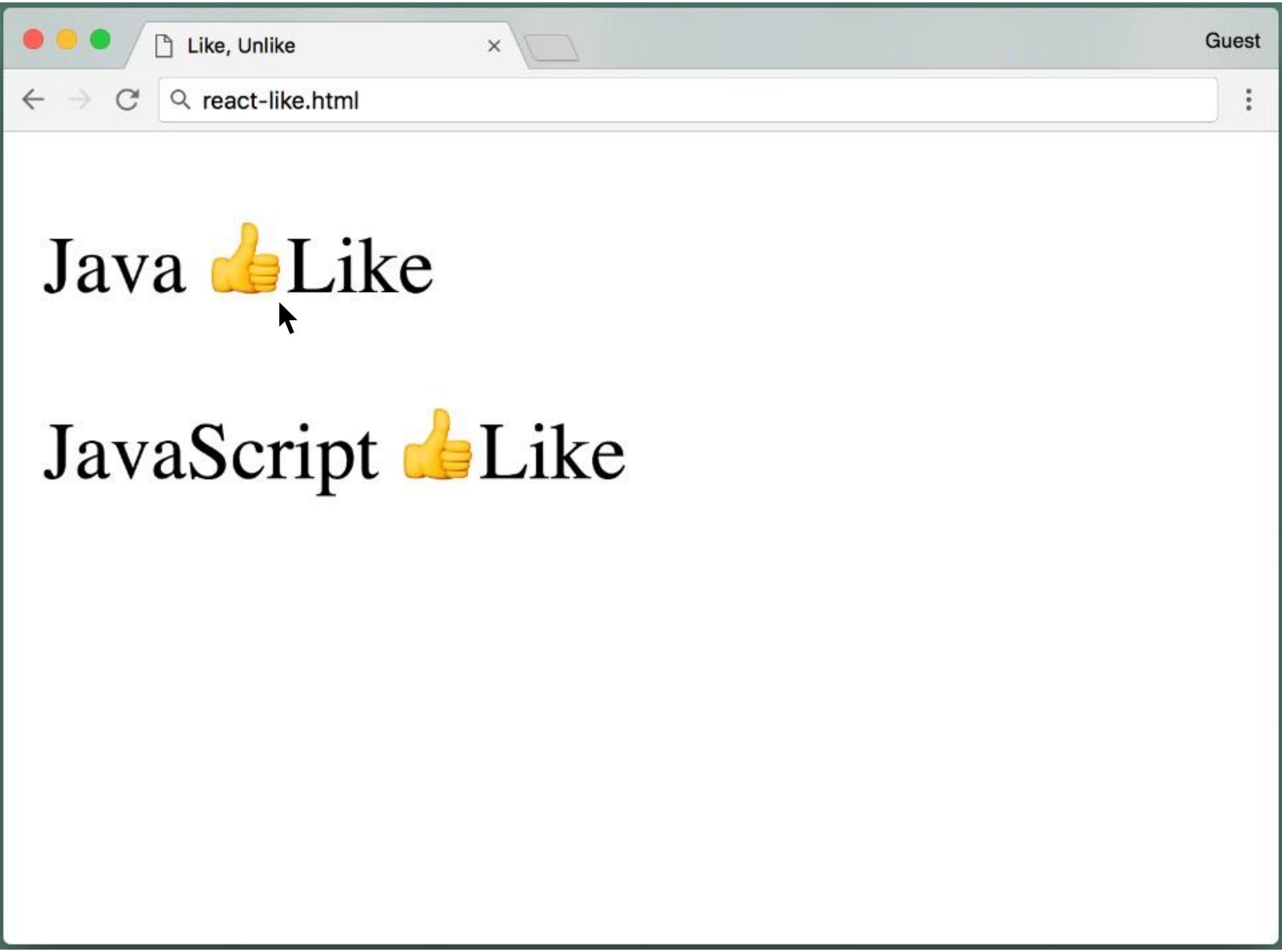
```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Counter extends React.Component {
           constructor(props) {
               super(props);
               this.state = { counts: 0 };
           }

           incrementCount() {
               this.setState({ counts: this.state.counts + 1 });
           }

           render() {
               return (
                   <div>
                      Count: {this.state.counts}
                      <button type="button" onClick={this.incrementCount.bind(this)}>
                           Increment
                      </button>
                   </div>
               );
           }
       };

    ReactDOM.
    createRoot(document.getElementById("container")).
    render(<Counter />);

   </script>
</body>
```



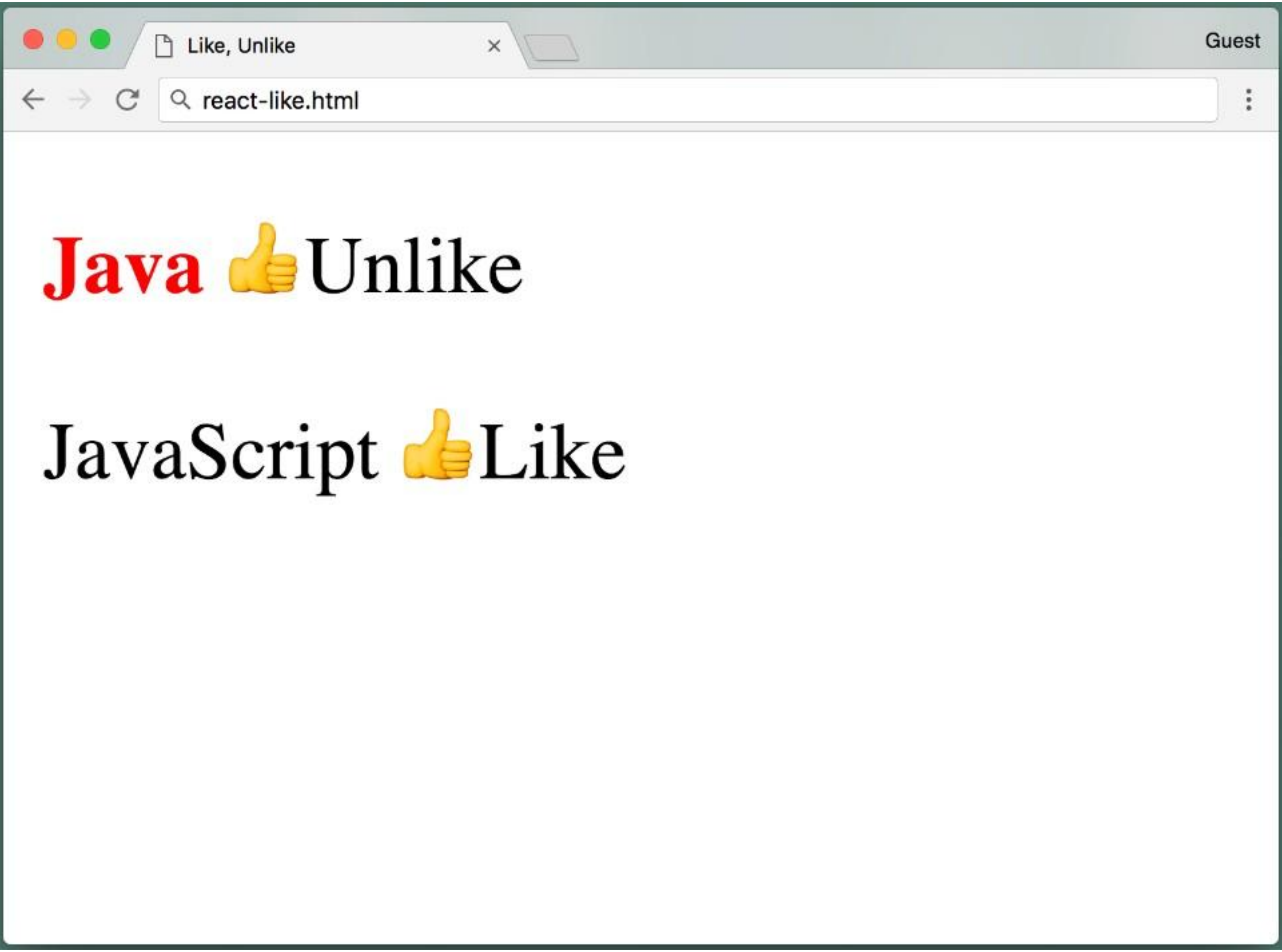
Java 👍Like

JavaScript 👍Like



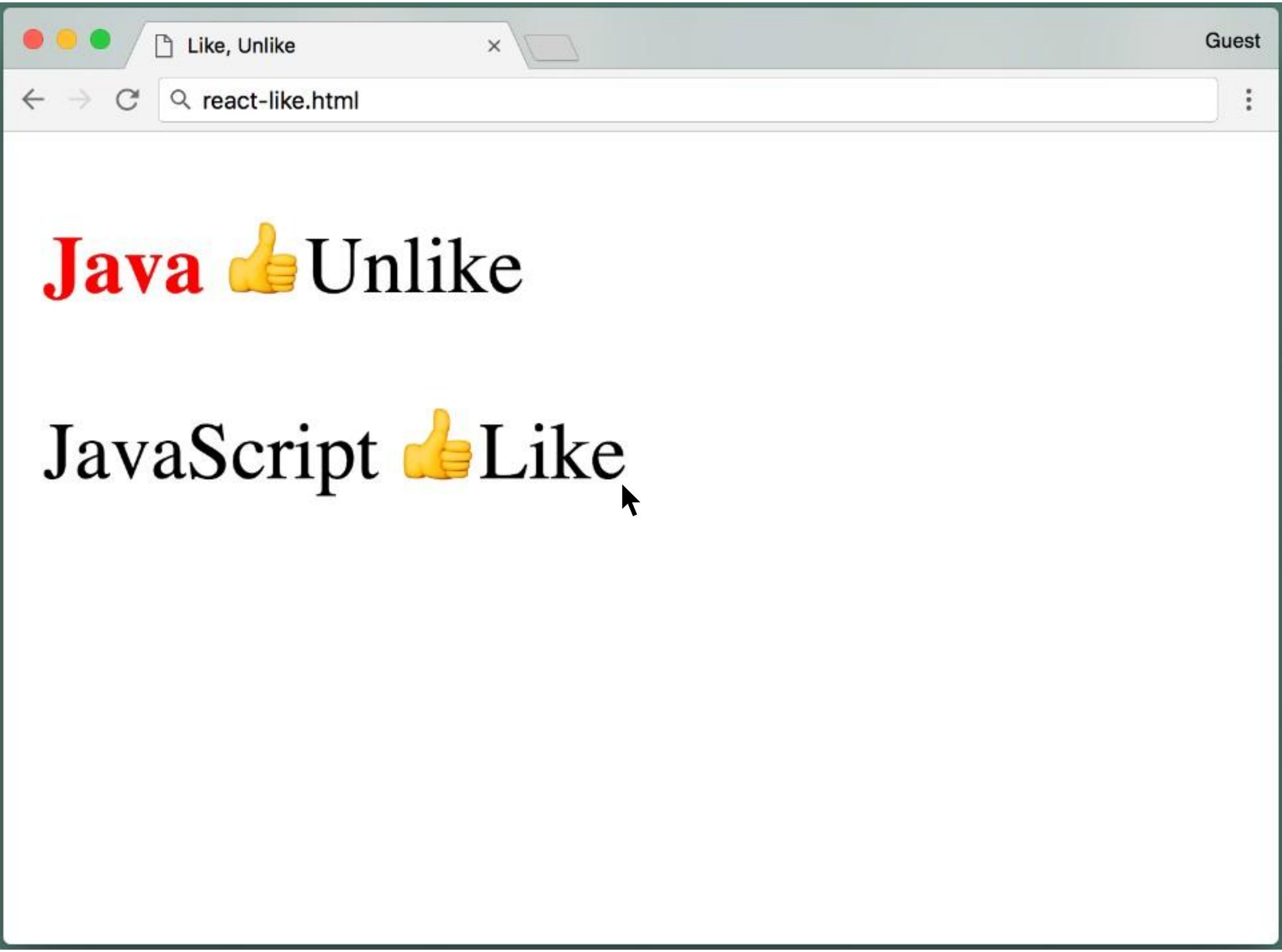
Java 👍Like

JavaScript 👍Like



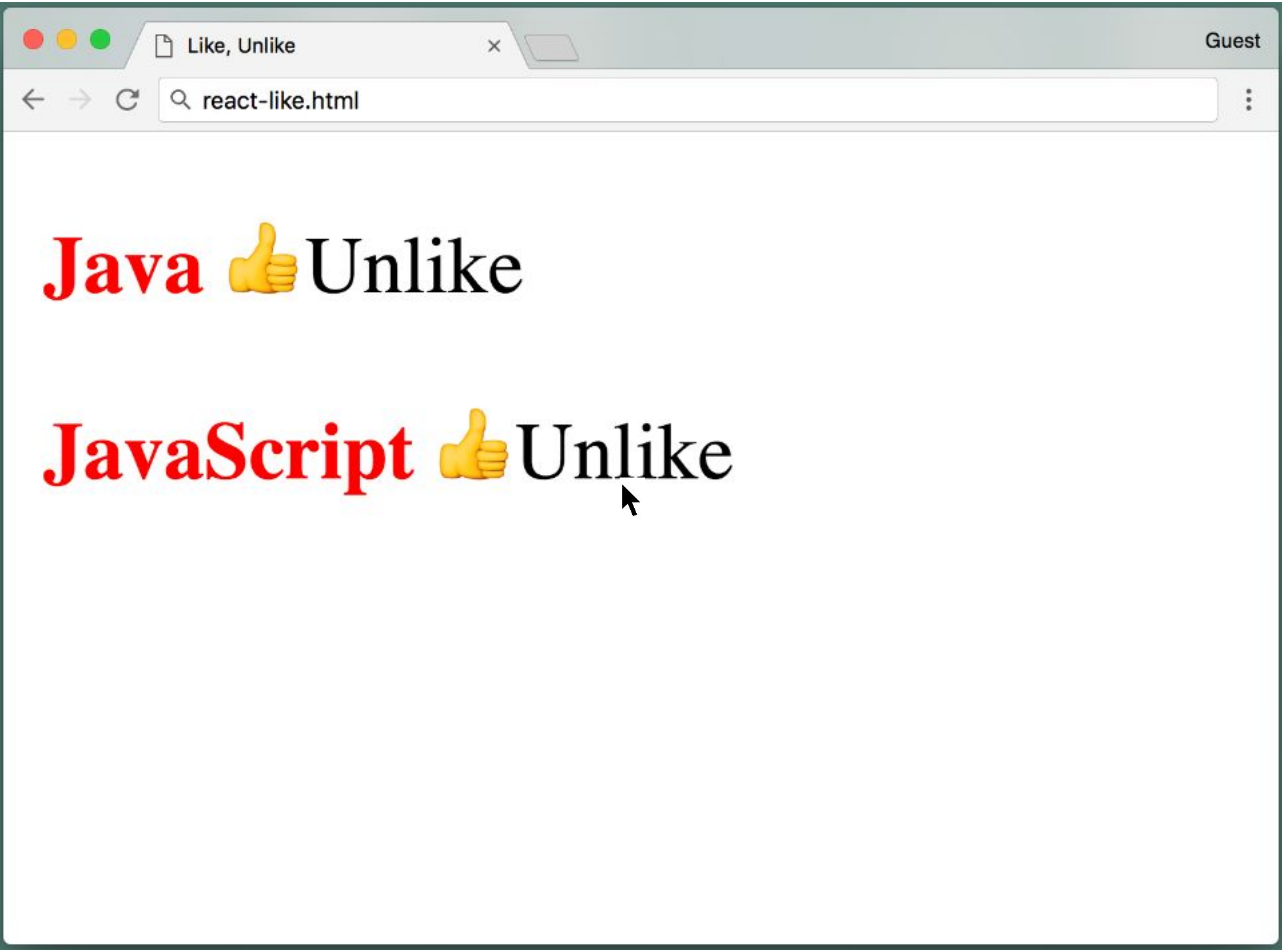
Java 👍Unlike

JavaScript 👍Like



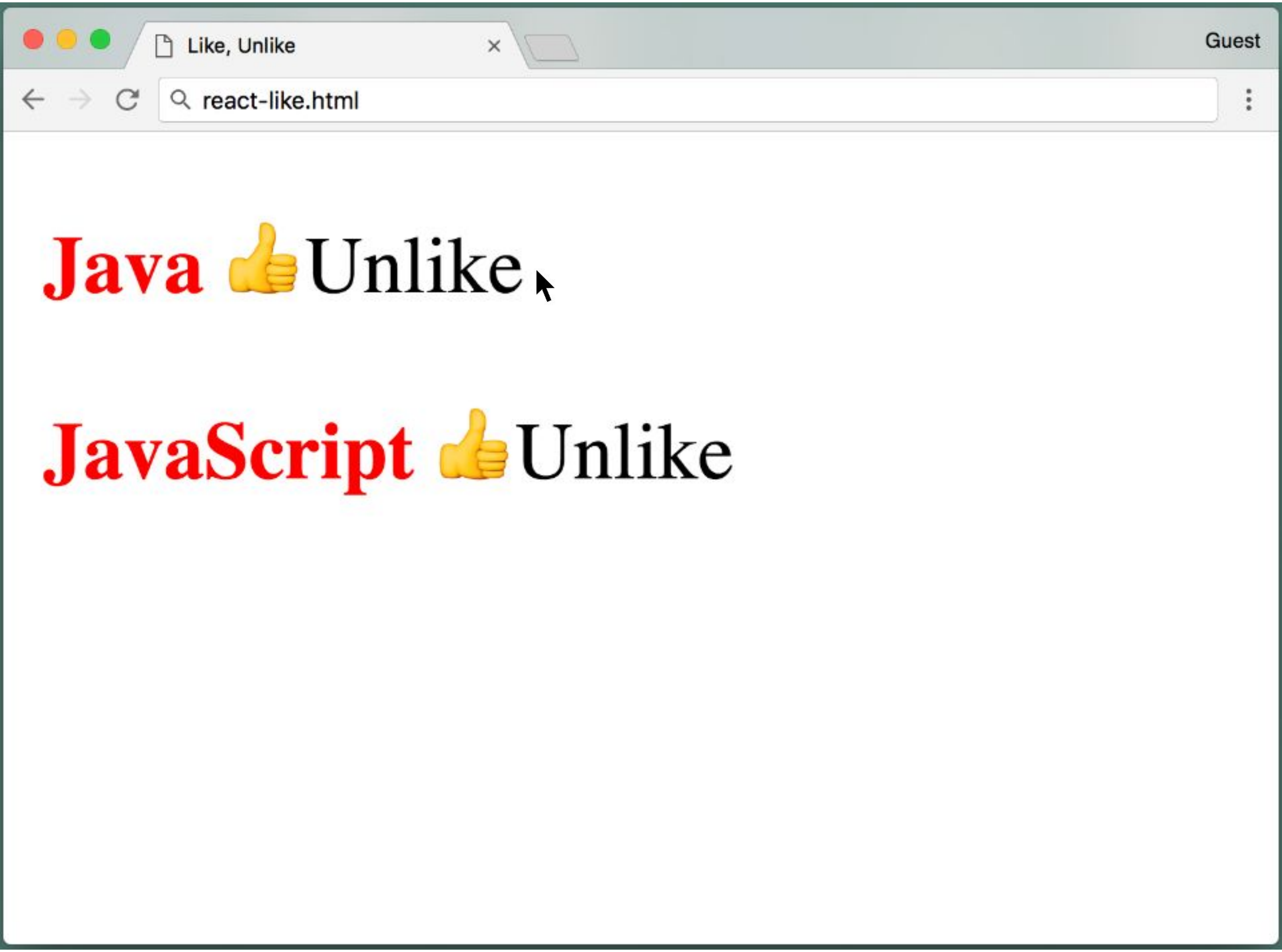
Java 👍Unlike

JavaScript 👍Like



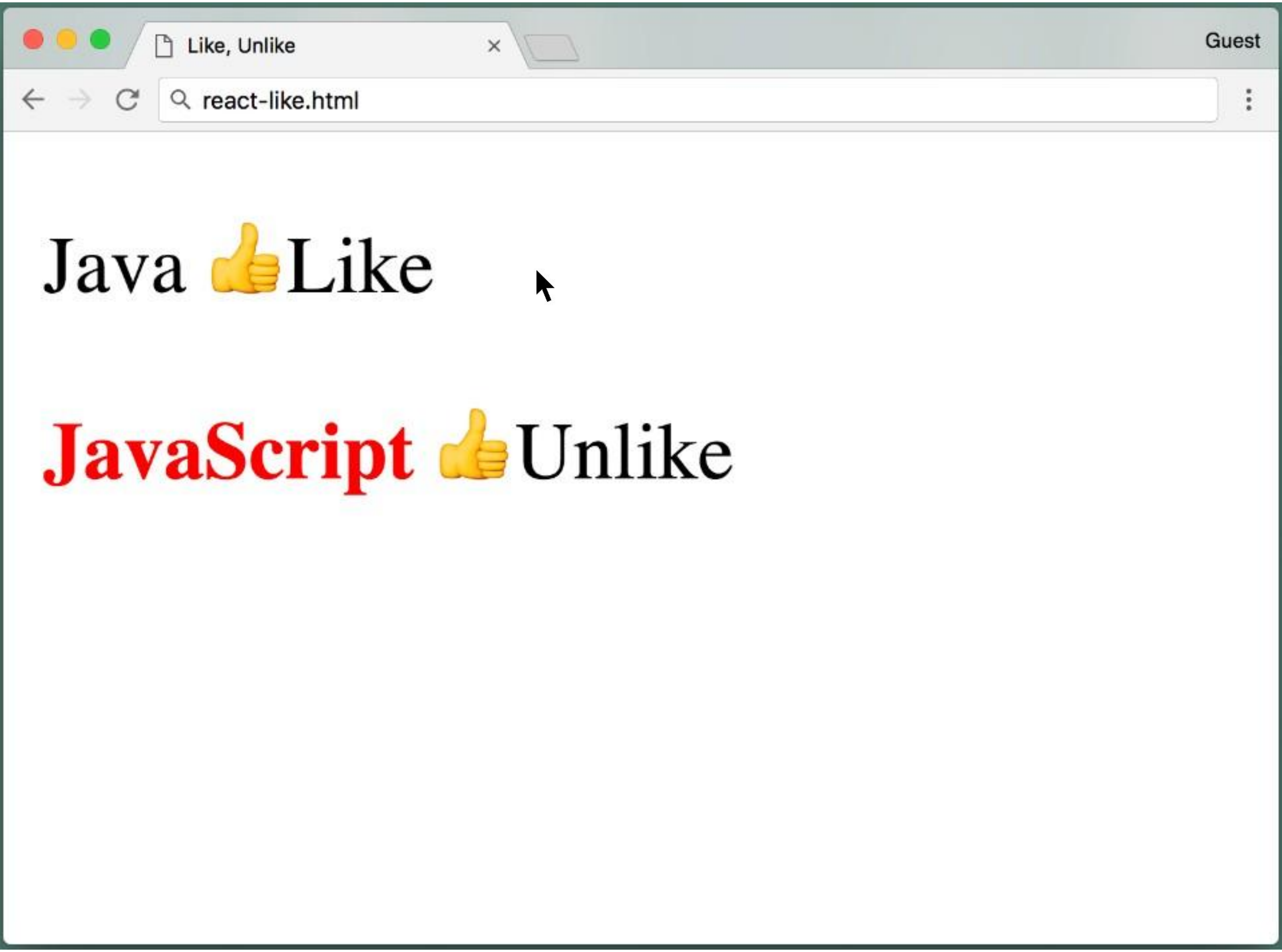
Java 👍Unlike

JavaScript 👍Unlike



Java 👍Unlike

JavaScript 👍Unlike



Java 👍Like

JavaScript 👍Unlike

```
<body>
   <div id="container"></div>

    <script type="text/babel">
       class Like extends React.Component {
           // implement Like class here
       };

    ReactDOM.createRoot(document.getElementById("container")).
    render(
        <div>
            <Like name="Java" />
             <Like name='JavaScript' />
         </div>);

   </script>
</body>
```

ex06.html

```
<body>
   <div id="container"></div>

    <script type="text/babel">
       class Like extends React.Component {
           // implement Like class here
       };

    ReactDOM.createRoot(document.getElementById("container")).
    render(
        <div>
            <Like name="Java" />
             <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

Define a component, and use it twice!

ex06.html

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props)
           {
               // initiate states
           }

           toggle(){
               // update state (setState)
           }

           render(){
               // invoke call back funciton (bind)
           }
       };
   ReactDOM.createRoot(document.getElementById("container")).
    render(
          <div>
               <Like name="Java" />
              <Like name='JavaScript' />
      </div>);
   </script>
</body>
```

ex06.html

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

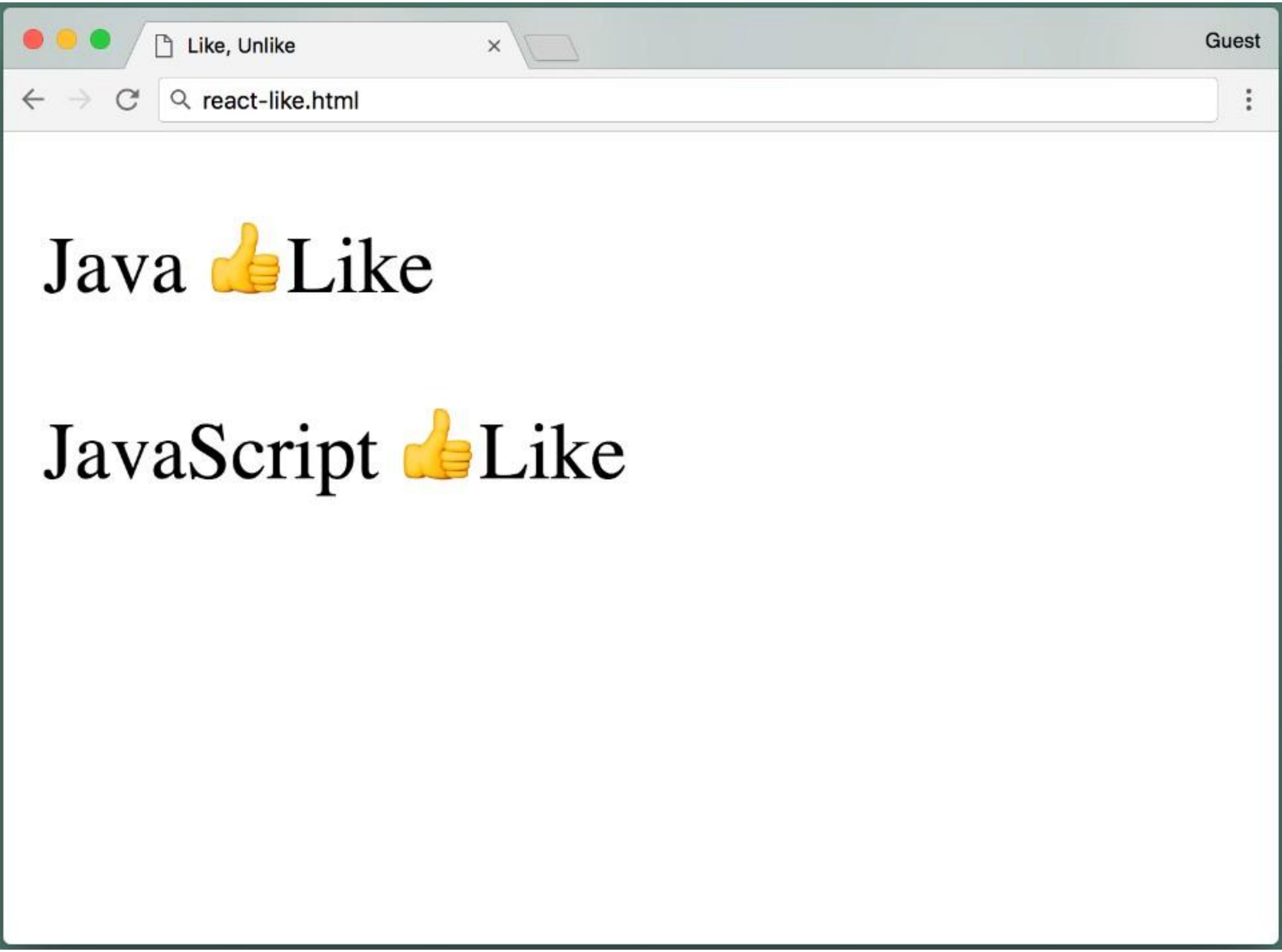
```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```



Java 👍Like

JavaScript 👍Like

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

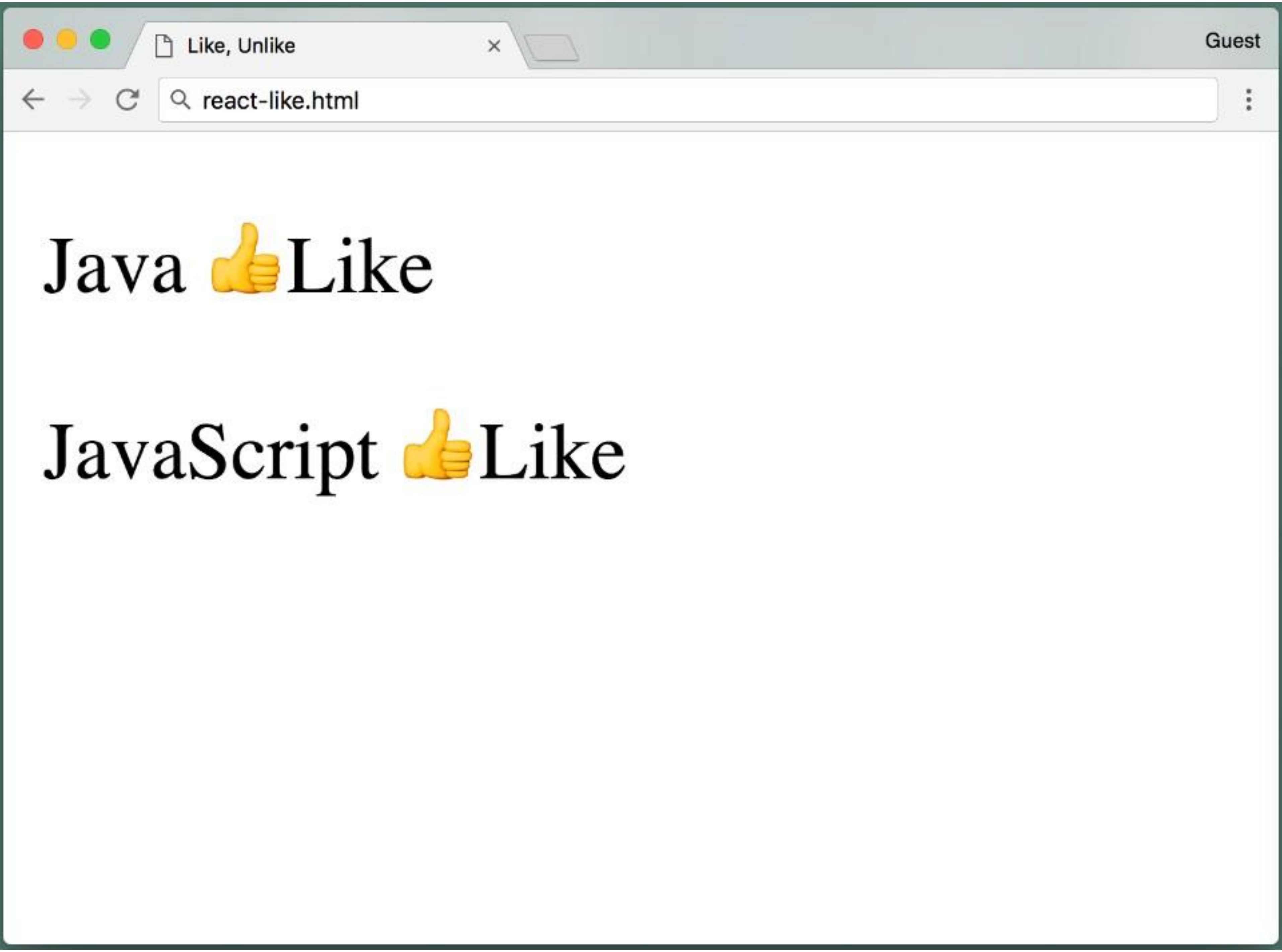
```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
          <div>
               <Like name="Java" />
               <Like name='JavaScript' />
          </div>);
   </script>
</body>
```

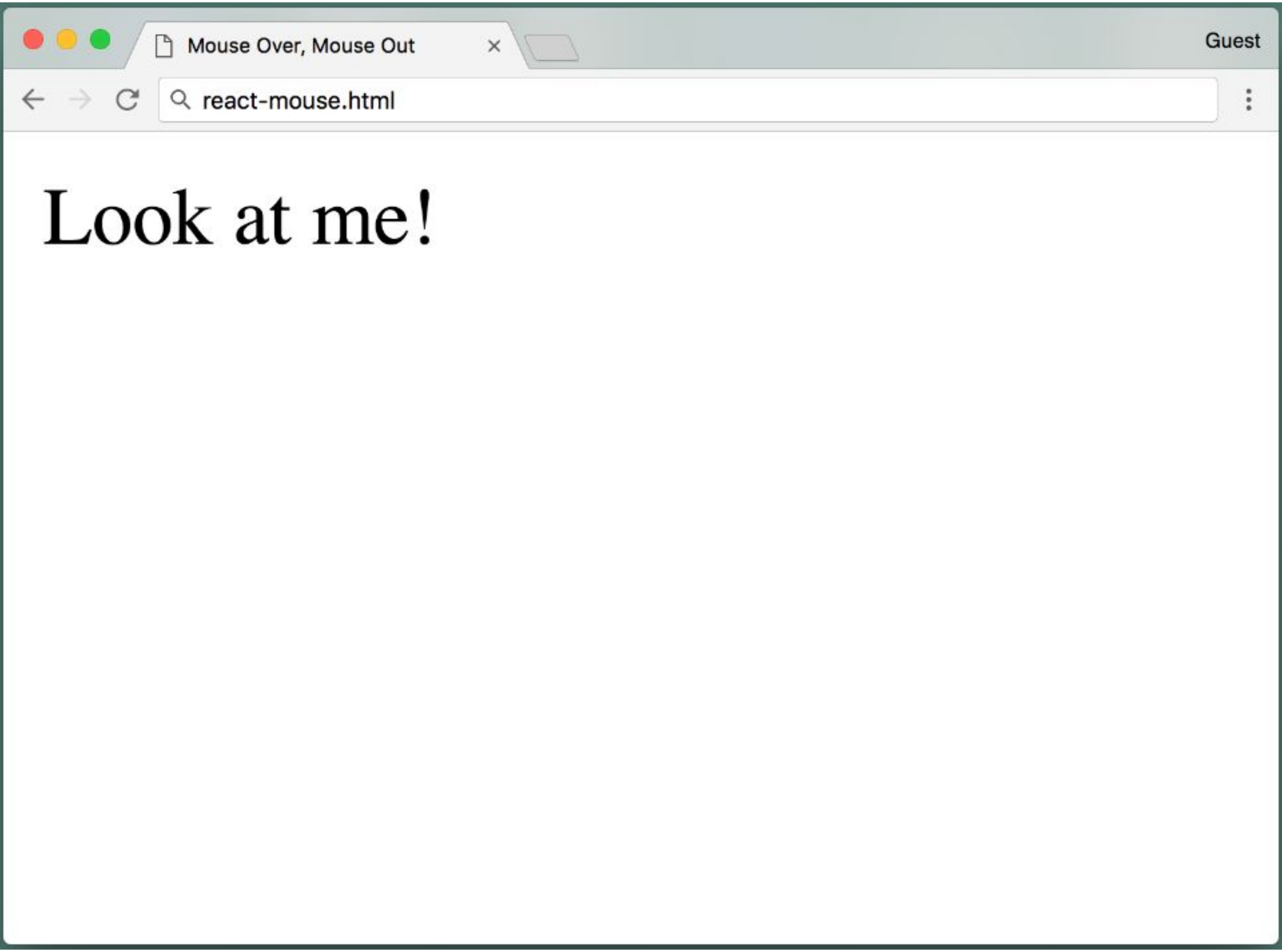
```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Like extends React.Component {
           constructor(props) {
               super(props);
               this.state = { liked: false };
           }

           toggle() {
               this.setState({ liked: !this.state.liked });}

           render() {
               var name = this.props.name;
               var txt = this.state.liked ? "Unlike" : "Like";
               var txtColor = this.state.liked ? "Red" : "Black";
               var txtWeight = this.state.liked ? "bold" : "normal";
               return (
                 <p>
                  <span style={{ color: txtColor, fontWeight: txtWeight}}>{name}</span>
                  <span onClick={this.toggle.bind(this)}> {'\ud83d\udc4d' + txt}</span>
                 </p>
               );
           }
       }

       ReactDOM.createRoot(document.getElementById("container")).render(
         <div>
              <Like name="Java" />
              <Like name='JavaScript' />
         </div>);
   </script>
</body>
```

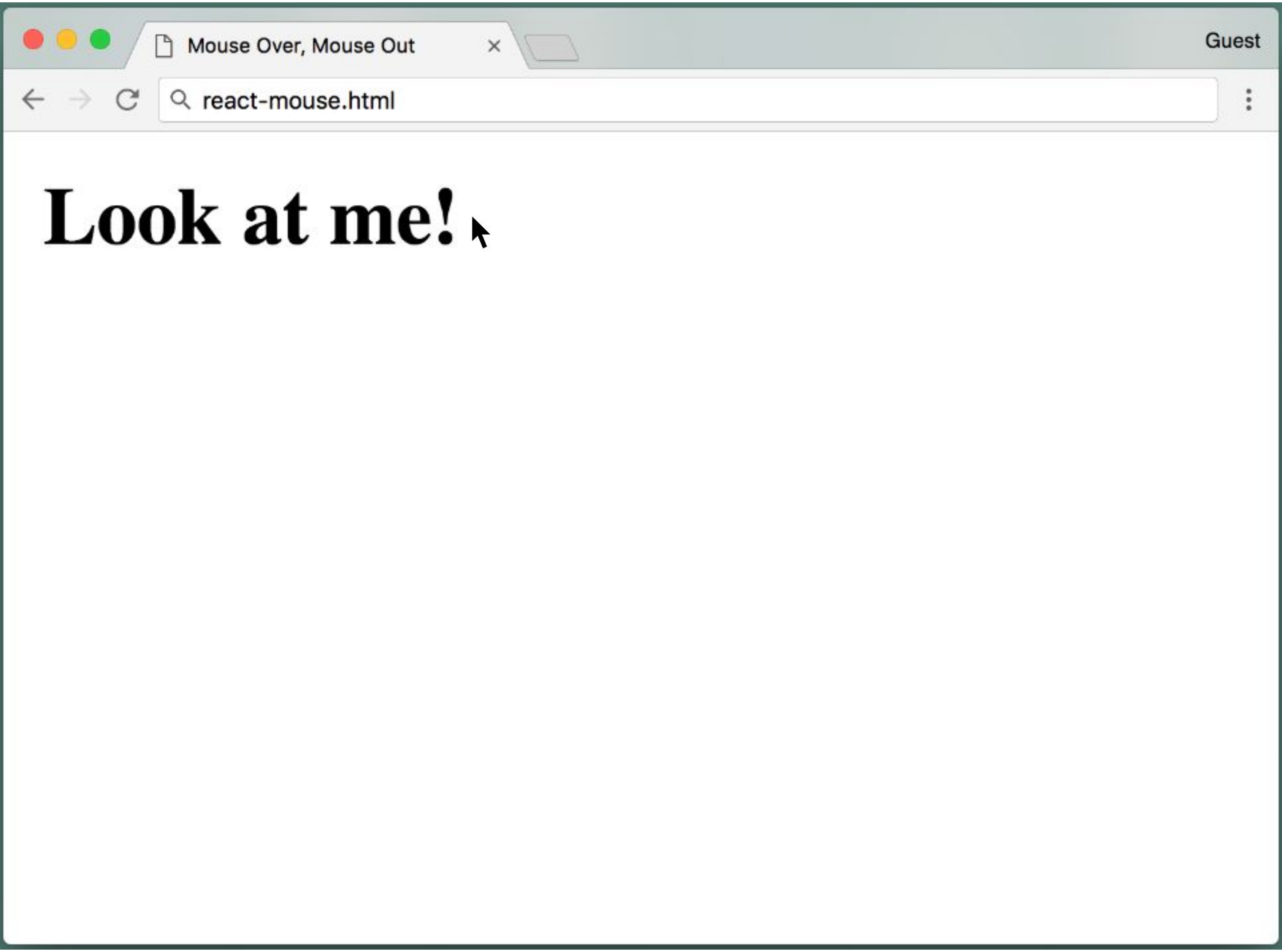


Java 👍Like

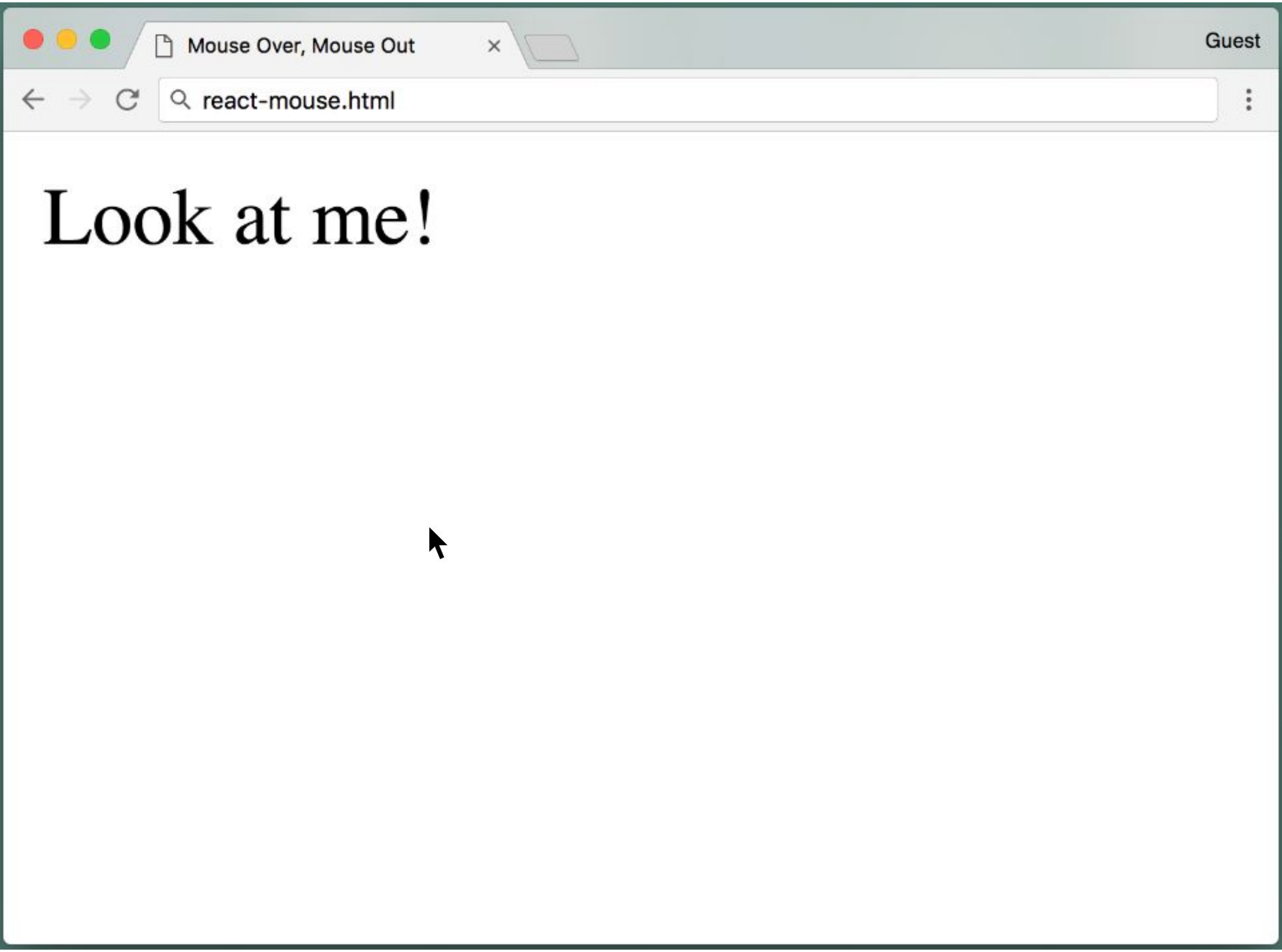
JavaScript 👍Like



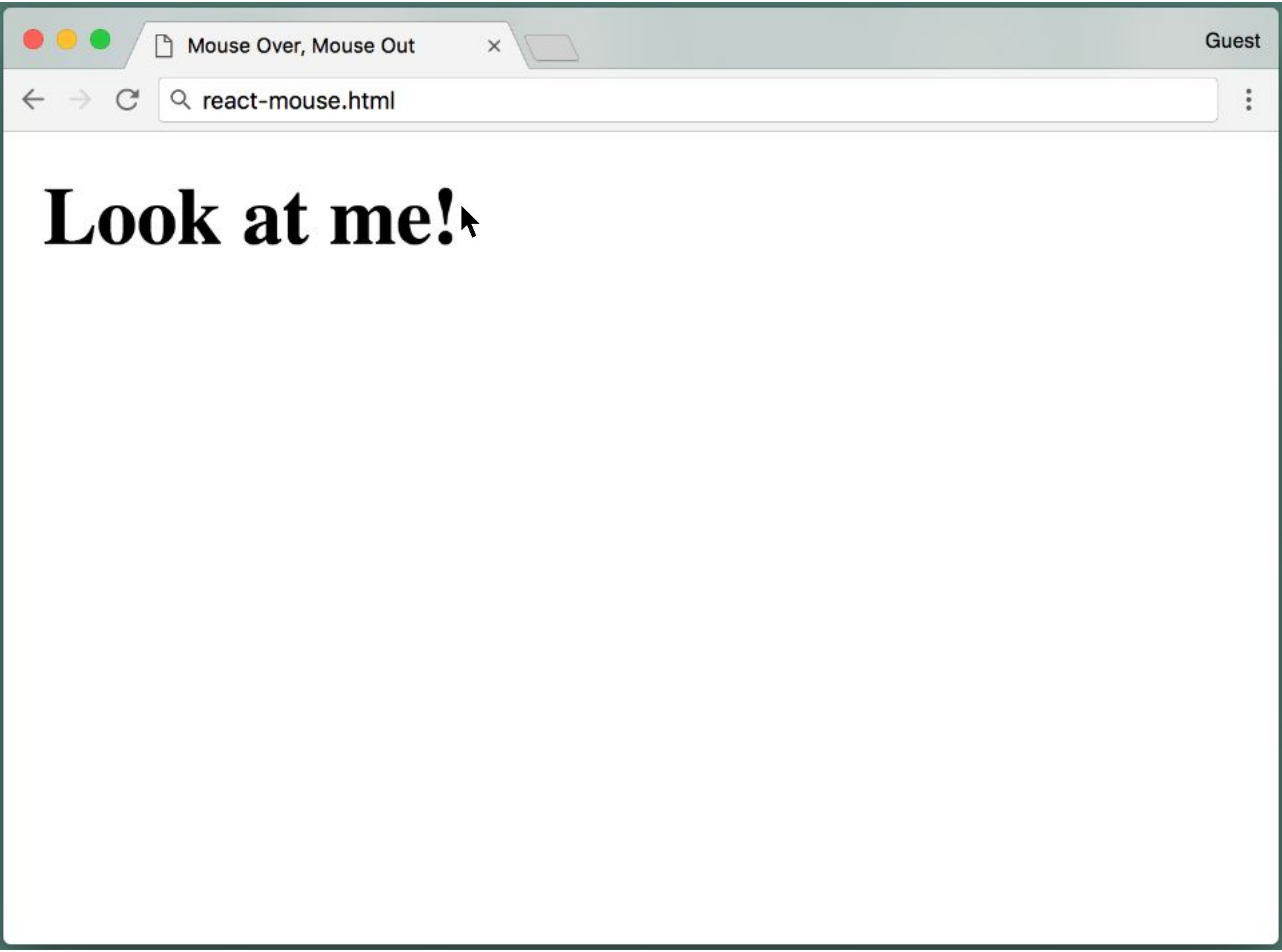
Look at me!



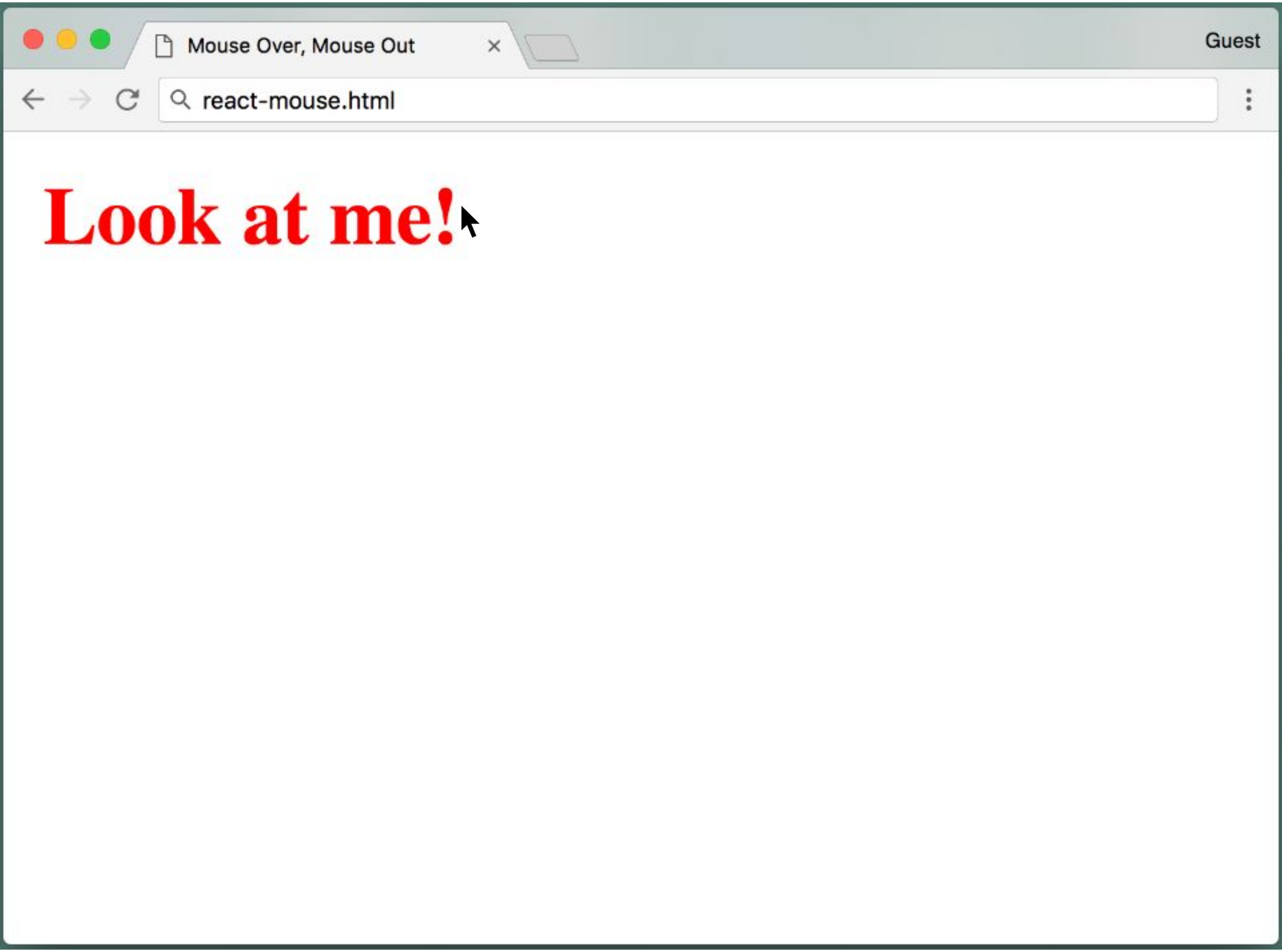
Look at me!



Look at me!



Look at me!



Look at me!

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Look extends React.Component{
           constructor(props) {
               super(props);
               // initiate states
           }
           handleMouseOver(){
               // set new states using setState
           }
           handleMouseOut(){
               // set new states using setState
           }
           handleClick(){
               // set new states using setState
           }

           render(){
               return(
                   // invoke call functions
               );
           }
       };

      ReactDOM.createRoot(document.getElementById("container")).render(
        <Look text="Look At ME" />
      );

   </script>
</body>
```

ex07.html

```
class Look extends React.Component {

        constructor(props) {
            super(props);
            this.state = { bold: false, color: "black" };
        }

        handleMouseOver() {
            this.setState({ bold: true });
        }

        handleMoveOut() {
            this.setState({ bold: false });
        }

        handleClick() {
            if (this.state.color == "black")
                this.setState({ color: "red" });
            else
                this.setState({ color: "black" });
        }

        render() {
            …
        }
    };
```

ex07.html

```
class Look extends React.Component {

           constructor(props) {
               super(props);
               this.state = { bold: false, color: "black" };
           }

           handleMouseOver() {
               this.setState({ bold: true });
           }

           handleMoveOut() {
               this.setState({ bold: false });
           }

           handleClick() {
               if (this.state.color == "black")
                   this.setState({ color: "red" });
               else
                   this.setState({ color: "black" });
           }

           render() {
              …
           }
       };
```

ex07.html

```
class Look extends React.Component {

      constructor(props) {
          super(props);
          this.state = { bold: false, color: "black" };
      }

      handleMouseOver() {
          this.setState({ bold: true });
      }

      handleMoveOut() {
          this.setState({ bold: false });
      }

      handleClick() {
          if (this.state.color == "black")
              this.setState({ color: "red" });
          else
              this.setState({ color: "black" });
      }

      render() {
          …
      }
  };
```

```
class Look extends React.Component {

           constructor(props) {
               super(props);
               this.state = { bold: false, color: "black" };
           }

           handleMouseOver() {
               this.setState({ bold: true });
           }

           handleMoveOut() {
               this.setState({ bold: false });
           }

           handleClick() {
               if (this.state.color == "black")
                   this.setState({ color: "red" });
               else
                   this.setState({ color: "black" });
           }

           render() {
              …
           }
       };
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
        <span style={{ color: txtColor, fontWeight: txtWeight }}
          onClick={this.handleClick.bind(this)}
          onMouseOver={this.handleMouseOver.bind(this)}
          onMouseOut={this.handleMoveOut.bind(this)}>
          {this.props.text}
         </span>
    );
}
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
         <span style={{ color: txtColor, fontWeight: txtWeight }}
           onClick={this.handleClick.bind(this)}
           onMouseOver={this.handleMouseOver.bind(this)}
           onMouseOut={this.handleMoveOut.bind(this)}>
           {this.props.text}
          </span>
    );
}
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
        <span style={{ color: txtColor, fontWeight: txtWeight }}
          onClick={this.handleClick.bind(this)}
          onMouseOver={this.handleMouseOver.bind(this)}
          onMouseOut={this.handleMoveOut.bind(this)}>
          {this.props.text}
         </span>
    );
}
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
        <span style={{ color: txtColor, fontWeight: txtWeight }}
          onClick={this.handleClick.bind(this)}
          onMouseOver={this.handleMouseOver.bind(this)}
          onMouseOut={this.handleMoveOut.bind(this)}>
          {this.props.text}
         </span>
    );
}
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
         <span style={{ color: txtColor, fontWeight: txtWeight }}
           onClick={this.handleClick.bind(this)}
           onMouseOver={this.handleMouseOver.bind(this)}
           onMouseOut={this.handleMoveOut.bind(this)}>
           {this.props.text}
          </span>
    );
}
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
        <span style={{ color: txtColor, fontWeight: txtWeight }}
          onClick={this.handleClick.bind(this)}
          onMouseOver={this.handleMouseOver.bind(this)}
          onMouseOut={this.handleMoveOut.bind(this)}>
          {this.props.text}
         </span>
    );
}
```

```
render() {
    var txtColor = this.state.color;
    var txtWeight = this.state.bold ? "bold" : "normal";

     return (
         <span style={{ color: txtColor, fontWeight: txtWeight }}
           onClick={this.handleClick.bind(this)}
           onMouseOver={this.handleMouseOver.bind(this)}
           onMouseOut={this.handleMoveOut.bind(this)}>
           {this.props.text}
          </span>
    );
}
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Look extends React.Component {

           constructor(props) {
               super(props);
               this.state = { bold: false, color: "black" };}

           handleMouseOver() {this.setState({ bold: true });}

           handleMoveOut() {this.setState({ bold: false });}

           handleClick() {
               if (this.state.color == "black")
                   this.setState({ color: "red" });
               else
                   this.setState({ color: "black" });}

           render() {
               var txtColor = this.state.color;
               var txtWeight = this.state.bold ? "bold" : "normal";
               return (
                   <span style={{ color: txtColor, fontWeight: txtWeight }}
                       onClick={this.handleClick.bind(this)}
                       onMouseOver={this.handleMouseOver.bind(this)}
                       onMouseOut={this.handleMoveOut.bind(this)}>
                       {this.props.text}
                   </span>
               );
           }
       };
      ReactDOM.createRoot(document.getElementById("container")).render(
        <Look text="Look At ME" />
      );

   </script>
</body>
```

```
<body>
   <div id="container"></div>
   <script type="text/babel">
       class Look extends React.Component {

           constructor(props) {
               super(props);
               this.state = { bold: false, color: "black" };}

           handleMouseOver() {this.setState({ bold: true });}

           handleMoveOut() {this.setState({ bold: false });}

           handleClick() {
               if (this.state.color == "black")
                   this.setState({ color: "red" });
               else
                   this.setState({ color: "black" });}

           render() {
               var txtColor = this.state.color;
               var txtWeight = this.state.bold ? "bold" : "normal";
               return (
                   <span style={{ color: txtColor, fontWeight: txtWeight }}
                       onClick={this.handleClick.bind(this)}
                       onMouseOver={this.handleMouseOver.bind(this)}
                       onMouseOut={this.handleMoveOut.bind(this)}>
                       {this.props.text}
                   </span>
               );
           }
       };
      ReactDOM.createRoot(document.getElementById("container")).render(
        <Look text="Look At ME" />
      );

   </script>
</body>
```

Summary

- We can bind user events in HTML elements to callback functions in React components

- When we invoke a component’s **`setState`** function, the **`render`** function will automatically be called and the component’s appearance can change accordingly