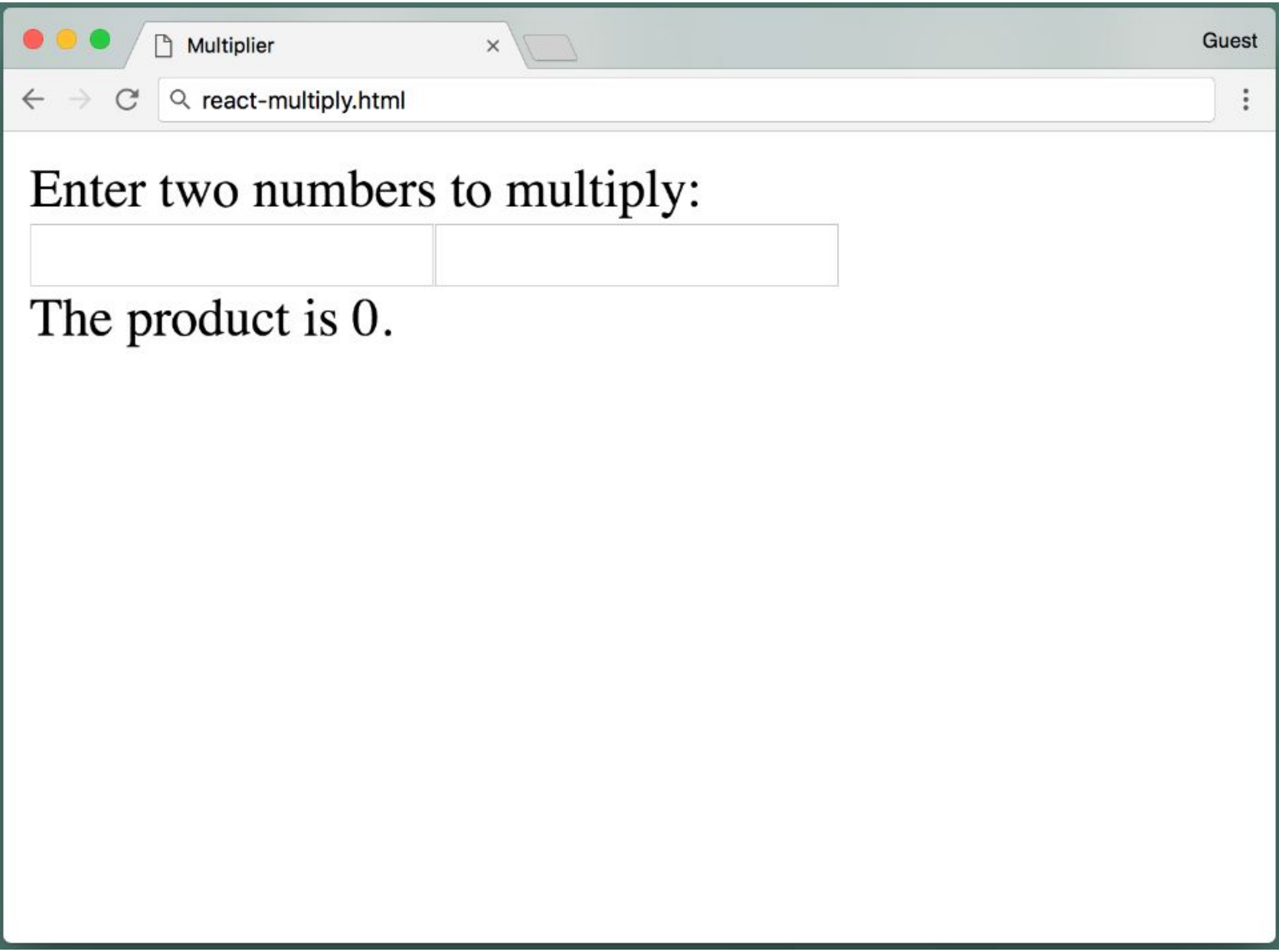
React Component Interaction Part 2

SENG 4640
Software Engineering for Web Apps
Winter 2023

Sina Keshvadi
Thompson Rivers University

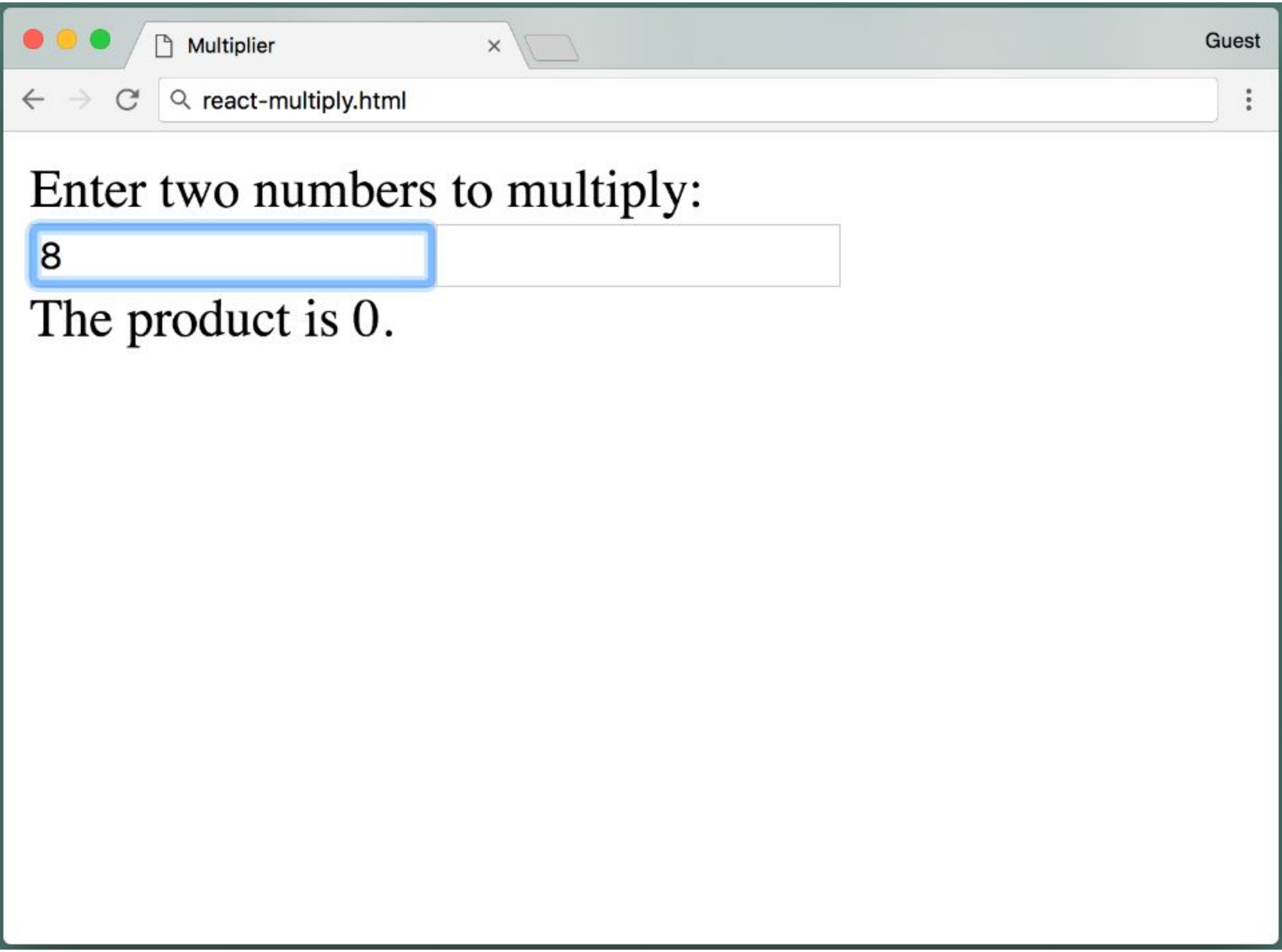
Review

- React allows us to create reusable, modularized components that can be combined to form web applications
- React handles re-rendering of components based on the structure of VirtualDOM



Enter two numbers to multiply:

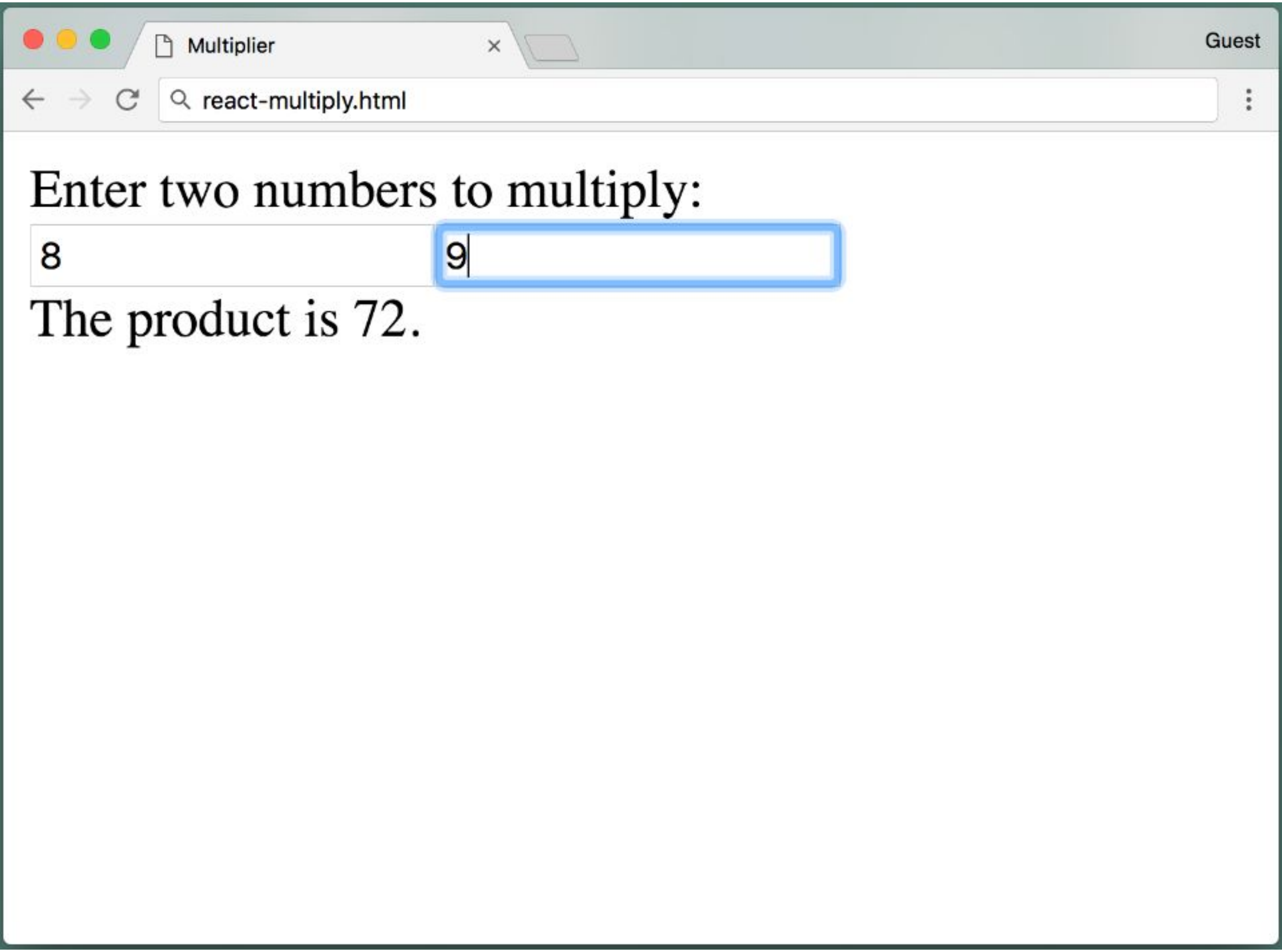
The product is 0.



Enter two numbers to multiply:

8

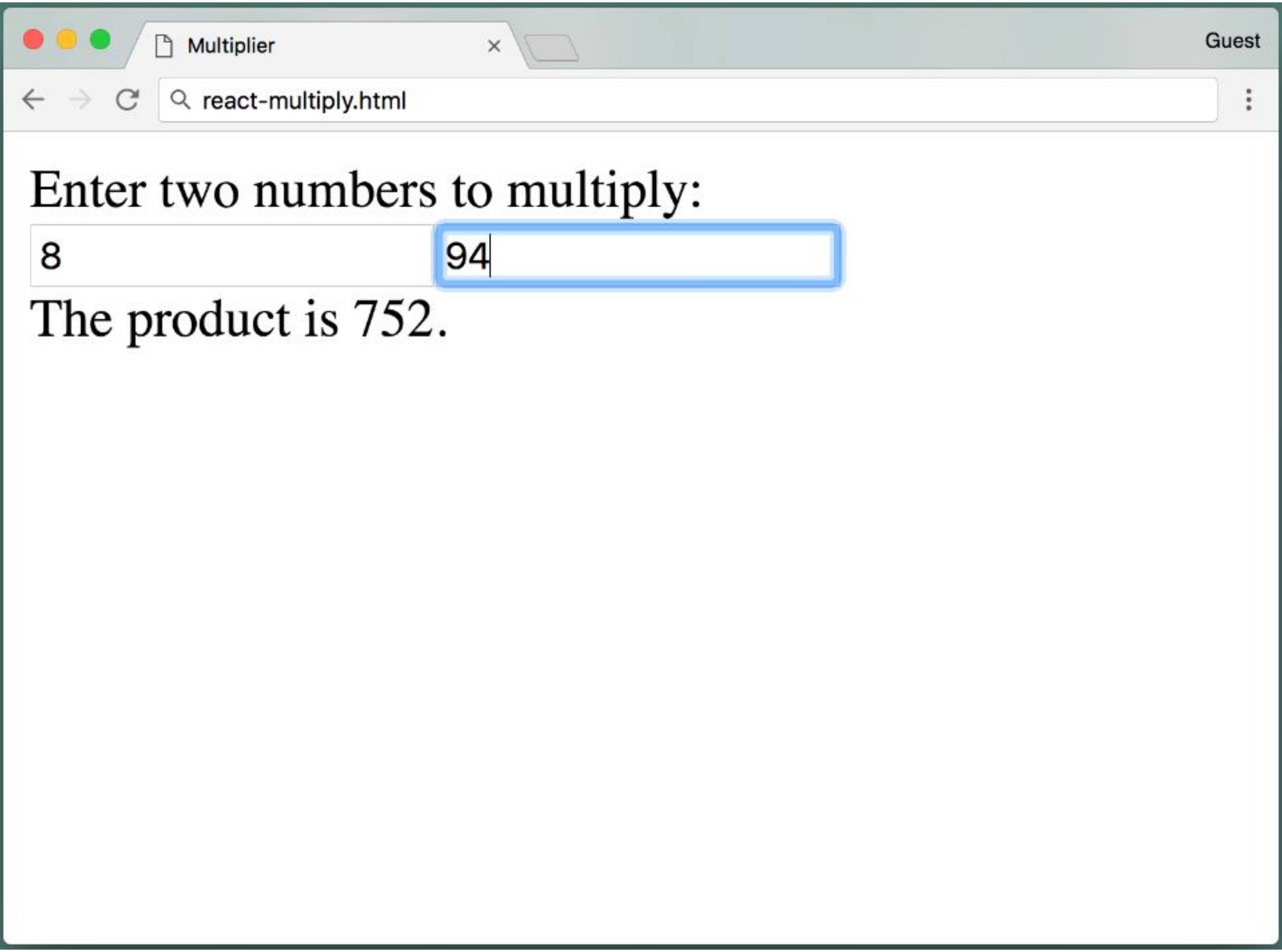
The product is 0.



Enter two numbers to multiply:

8 9

The product is 72.

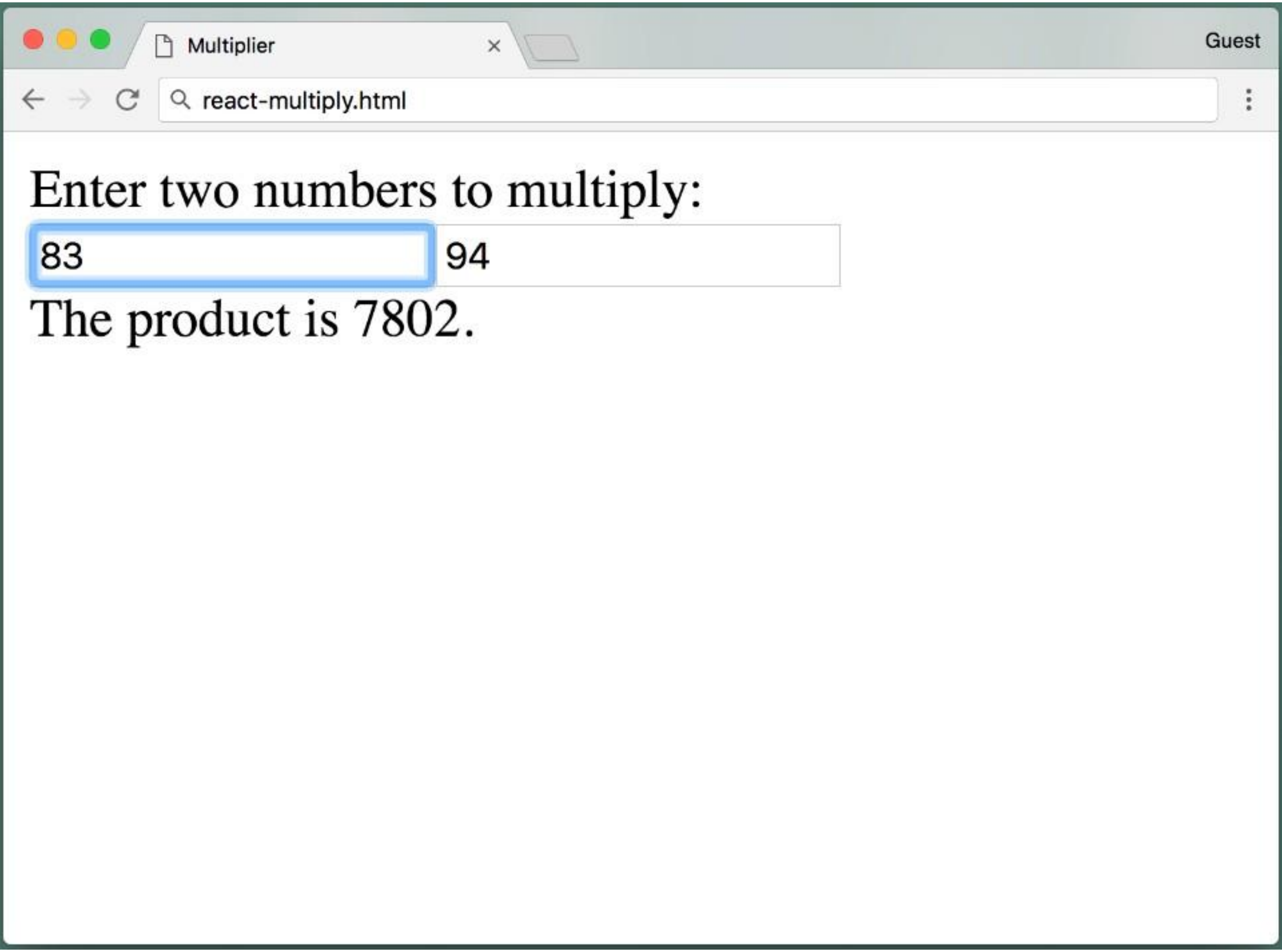


Enter two numbers to multiply:

8

94

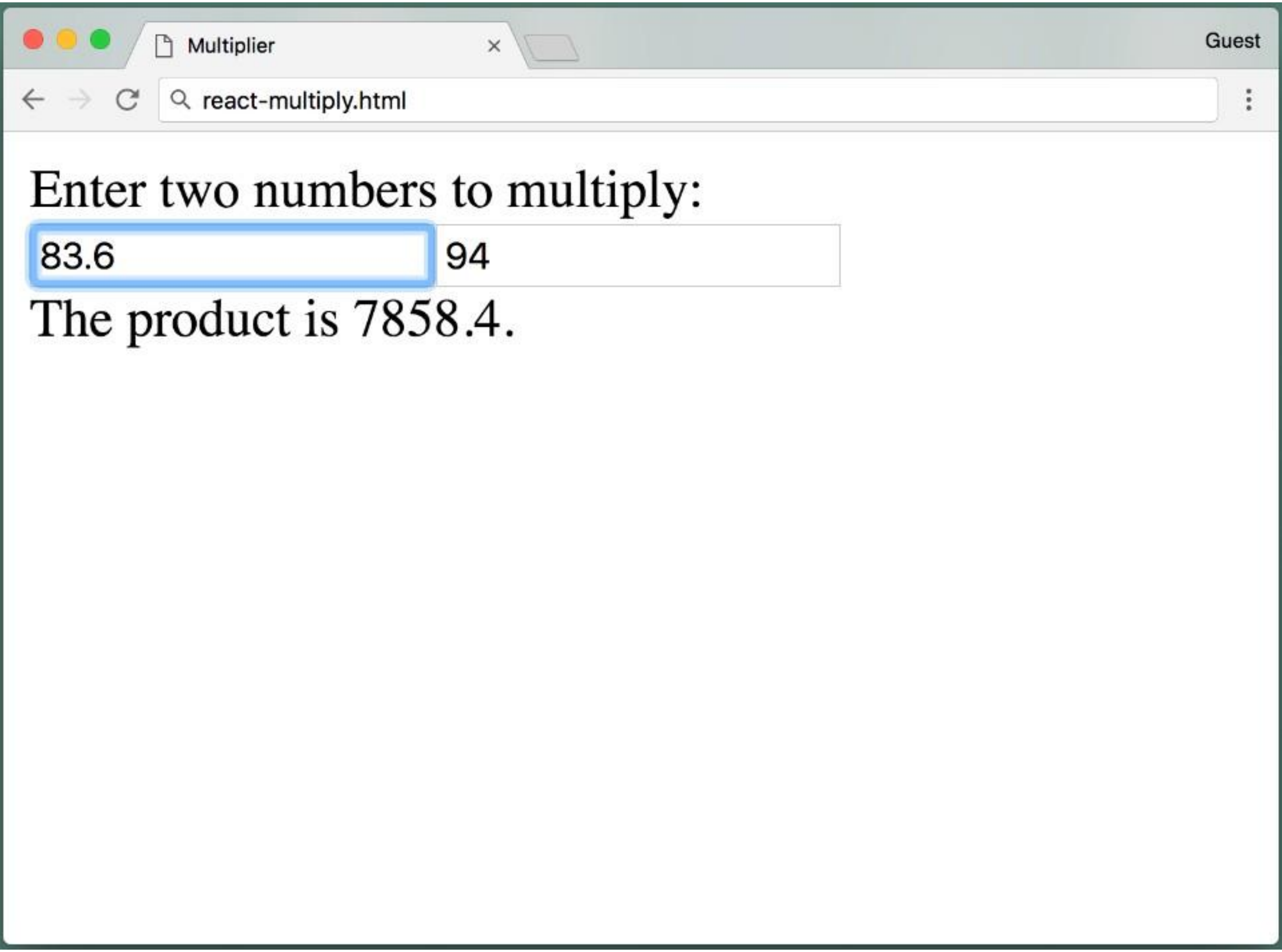
The product is 752.



Enter two numbers to multiply:

83 94

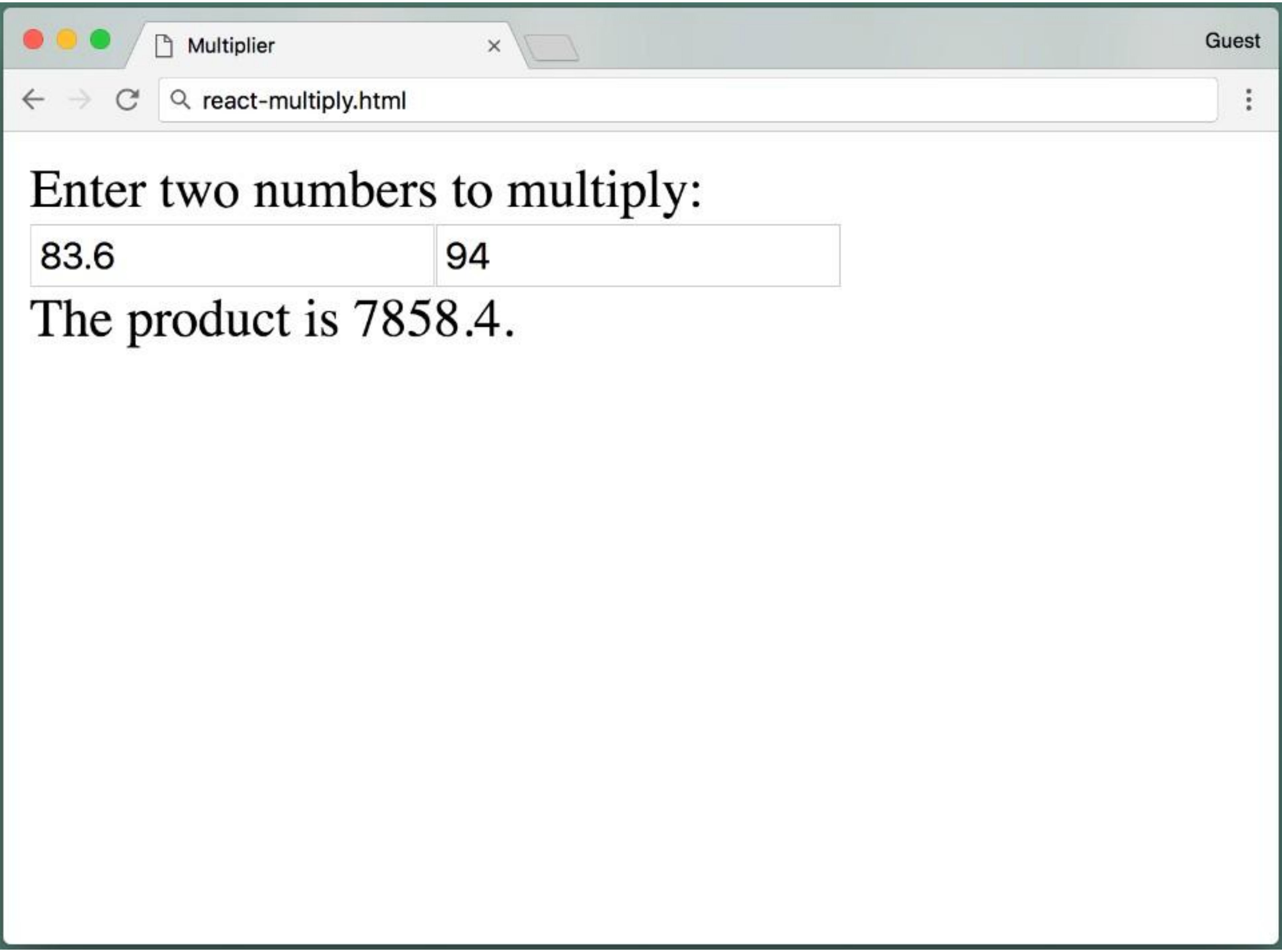
The product is 7802.



Enter two numbers to multiply:

83.6 94

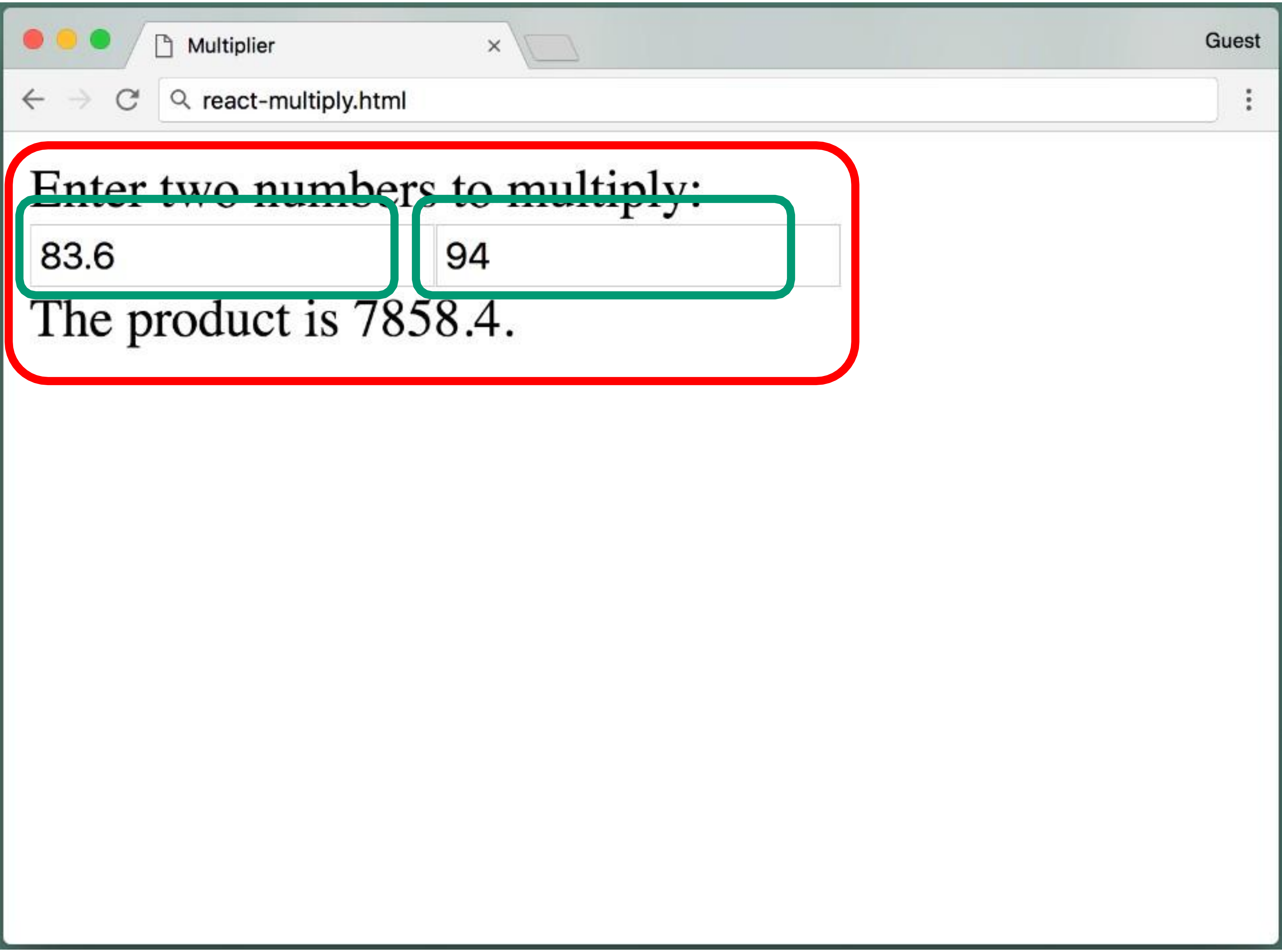
The product is 7858.4.



Enter two numbers to multiply:

83.6 | 94

The product is 7858.4.

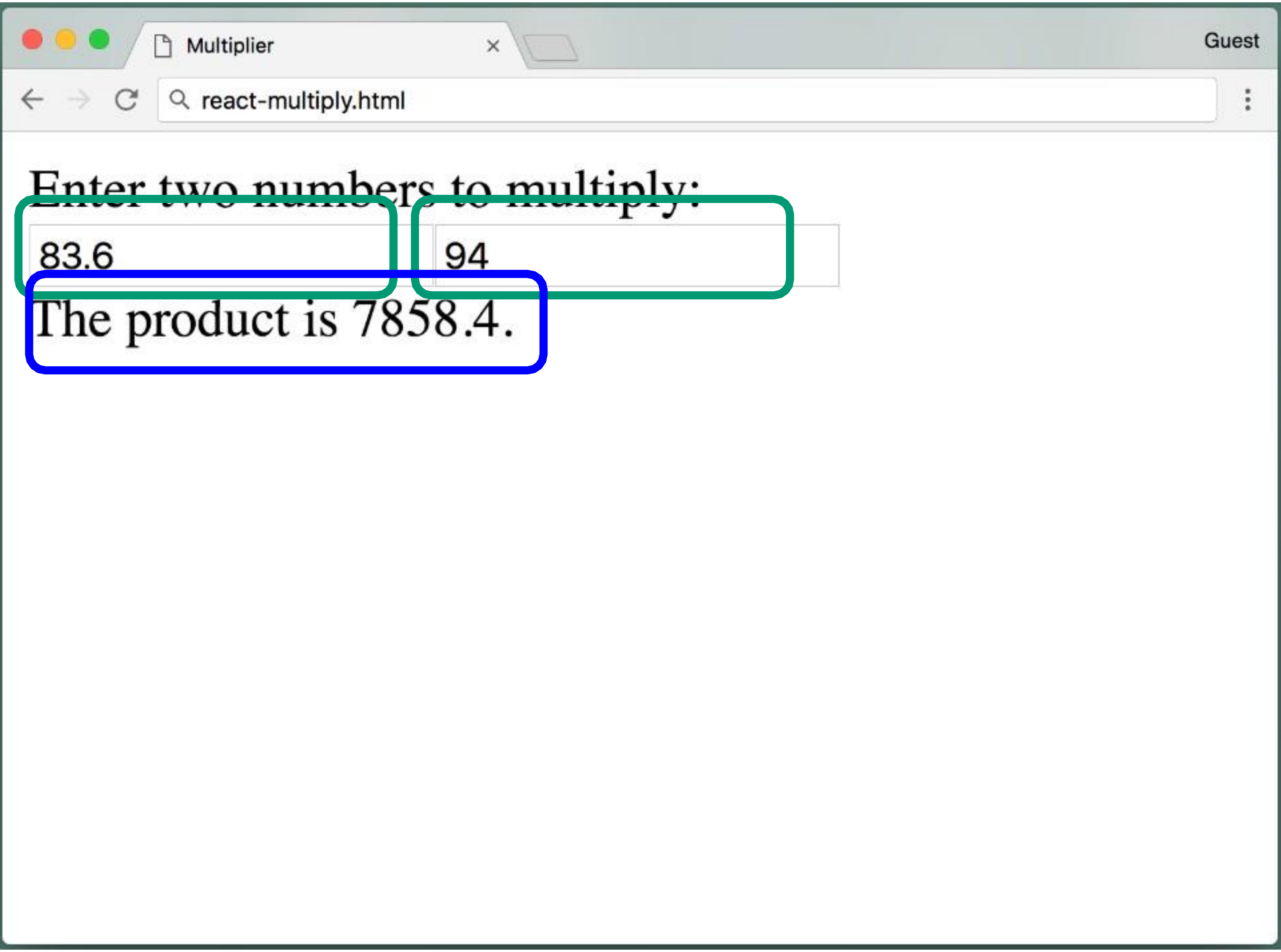


Enter two numbers to multiply:

83.6

94

The product is 7858.4.

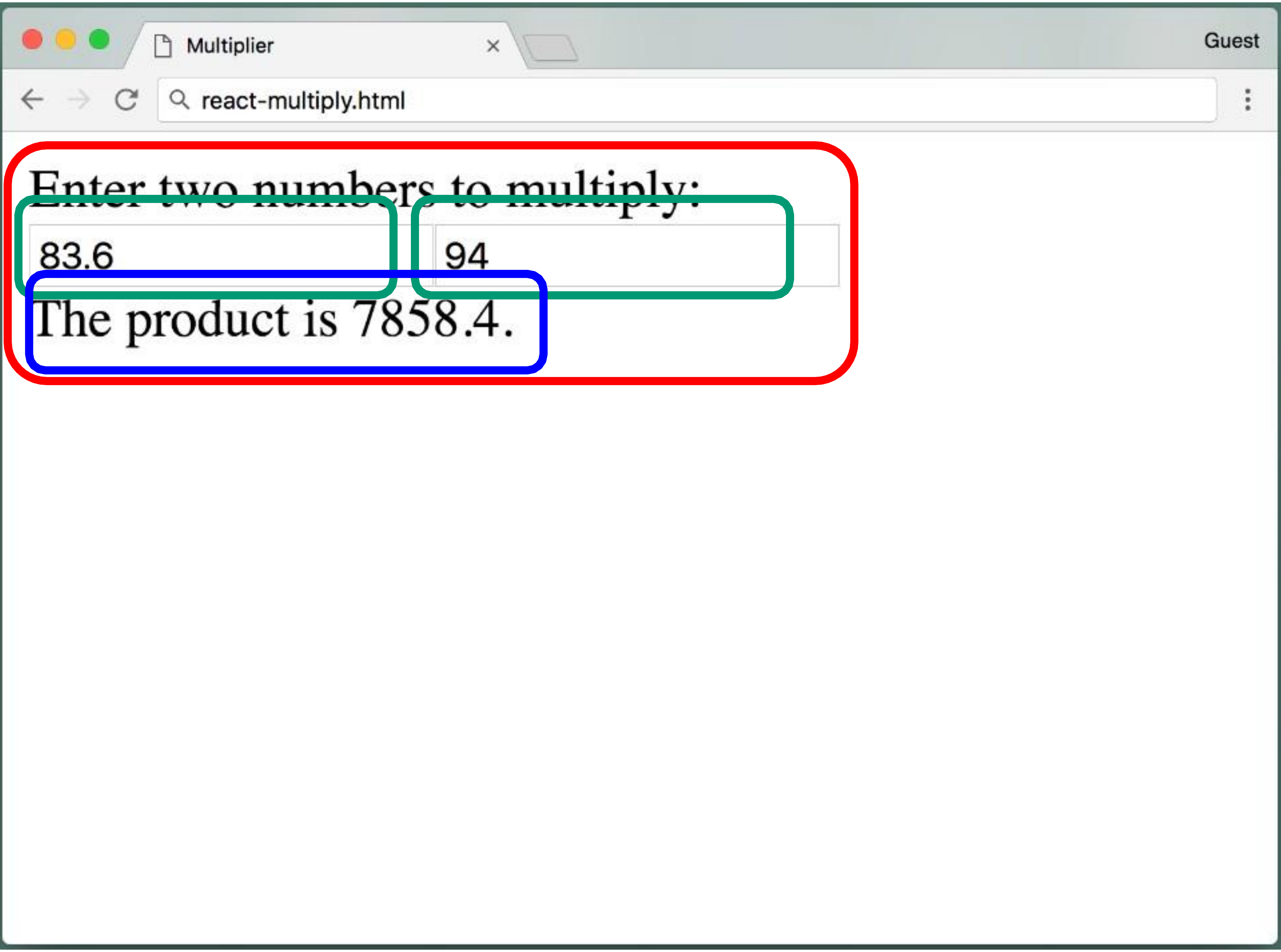


Enter two numbers to multiply:

83.6

94

The product is 7858.4.



Enter two numbers to multiply:

83.6

94

The product is 7858.4.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .                                                                   ex10.html
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

ex10.html

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .                                                                  ex10.html
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };
. . .
```

Create a multiply property and setting it to its multiply function using bind

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };
. . .
```

the id of the input box that's being changed and the value that is in that box.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

If the id is 1, that means that this is input box number 1

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

The action is a property that's passed to the number input field.

Passing a function from a parent to its child

- The action is a property that's passed to the number input field.
- But what's different about this?
- In past examples, the property has always been some sort of variable.
- We initialized the string, we initialized an item, but here, we're initializing it with a function.
- That is we're passing a function from the multiplier, the parent, to its child, the NumberInputField.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
        constructor(props) {
            super(props);
            this.state = { input1: 0, input2: 0, product: 0 };
            this.multiply = this.multiply.bind(this);
        }
        multiply(id, val) {
            if (id == 1) {
                this.setState({input1: val, product: val * this.state.input2});
            }
            else if (id == 2) {
                this.setState({input2: val, product: this.state.input1 * val});
            }
        }
        render() {
            return (
                <div>
                    <NumberInputField id="1" action={this.multiply} />
                    <NumberInputField id="2" action={this.multiply} />
                    <OutputField product={this.state.product} />
                </div>
            );
        }
    };

. . .
```

The property of the outputField will be the product set to the products that we've calculated within the multiplier.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

we bind the handleChange function to a handleChange variable that we can later use.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

It invokes it's props that were set when this component was created.

- It accesses it's props that were set when this component was created and invoke this action function.
- So action is part of the props and was set when this component was created to be the multiply function in the multiplier component.
- That's how this component, NumberInputField, can call a function can in another component because that function was passed to it as its props.
- When it calls that function, it passes its own ID and the value that's in the input box.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      multiply(id, val) {...}
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
     ...
    };
================================================================================
class NumberInputField extends React.Component {
    constructor(props) {
        super(props);
        this.handleChange = this.handleChange.bind(this);
    }

    handleChange(e) {
        this.props.action(this.props.id, e.target.value);
    }
    render() {
        return (
            <input onChange={this.handleChange}></input>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
      <NumberInputField id="2" action={this.multiply} />
      <OutputField product={this.state.product} />
     ...
    };
================================================================================
class OutputField extends React.Component {
    render() {
        return (
            <div>The product is {this.props.product}.
            </div>
        );
    }
};
```

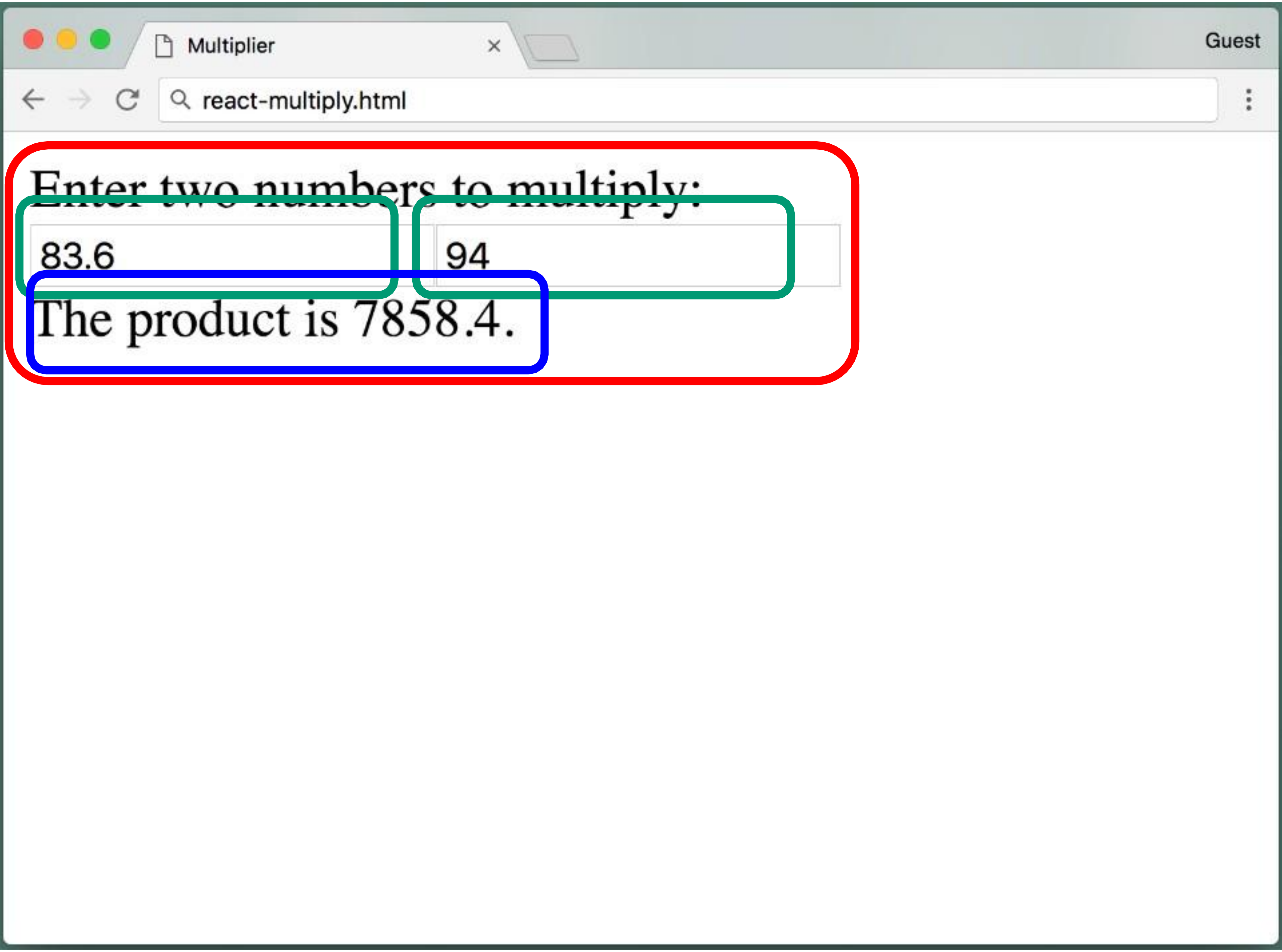
```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
      <NumberInputField id="2" action={this.multiply} />
      <OutputField product={this.state.product} />
     ...
    };
================================================================================
class OutputField extends React.Component {
    render() {
        return (
            <div>The product is {this.props.product}.
            </div>
        );
    }
};
```

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
      <NumberInputField id="2" action={this.multiply} />
      <OutputField product={this.state.product} />
     ...
    };
================================================================================
class OutputField extends React.Component {
    render() {
        return (
            <div>The product is {this.props.product}.
            </div>
        );
    }
};
```

The text that reads the product is with the product that was passed to it.

```
<div id="container"></div>
<script type="text/babel">
    class Multiplier extends React.Component {
     ...
      <NumberInputField id="1" action={this.multiply} />
      <NumberInputField id="2" action={this.multiply} />
      <OutputField product={this.state.product} />
     ...
    };
================================================================================
class OutputField extends React.Component {
    render() {
        return (
            <div>The product is {this.props.product}.
            </div>
        );
    }
};

     ReactDOM.createRoot(document.getElementById('container'))
     .render(<Multiplier />);
   </script>
```



Enter two numbers to multiply:

83.6

94

The product is 7858.4.

Review

- React allows us to create reusable, modularized components that can be combined to form web applications

- Components can communicate with each other via callback methods that are set as props