Working with Components in React 18

Implementing Components in React

Peter Kellner

Developer and Author

peterkellner.net | linkedin.com/in/peterkellner99
Mastodon: @pkellner@peterkellner.net

Working with Components in React 18

Version Check

Version Check

This course was created by using React 18

Version Check

Version Check

This course is 100% applicable to React 18

What are Components?

Independent bits of code

Often designed to be reusable

In React

- Render HTML
- Render other components

What are Components?

Independent bits of code

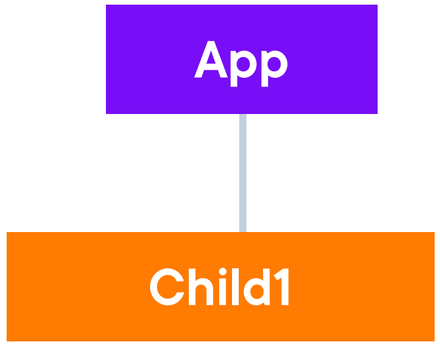
Often designed to be reusable

In React

- Render HTML
- Render other components

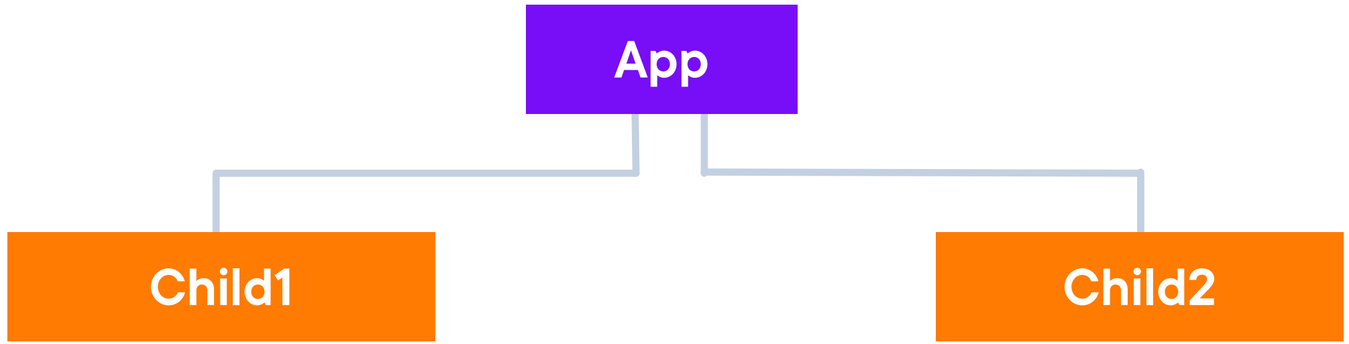
React Component Hierarchy

React Component Hierarchy

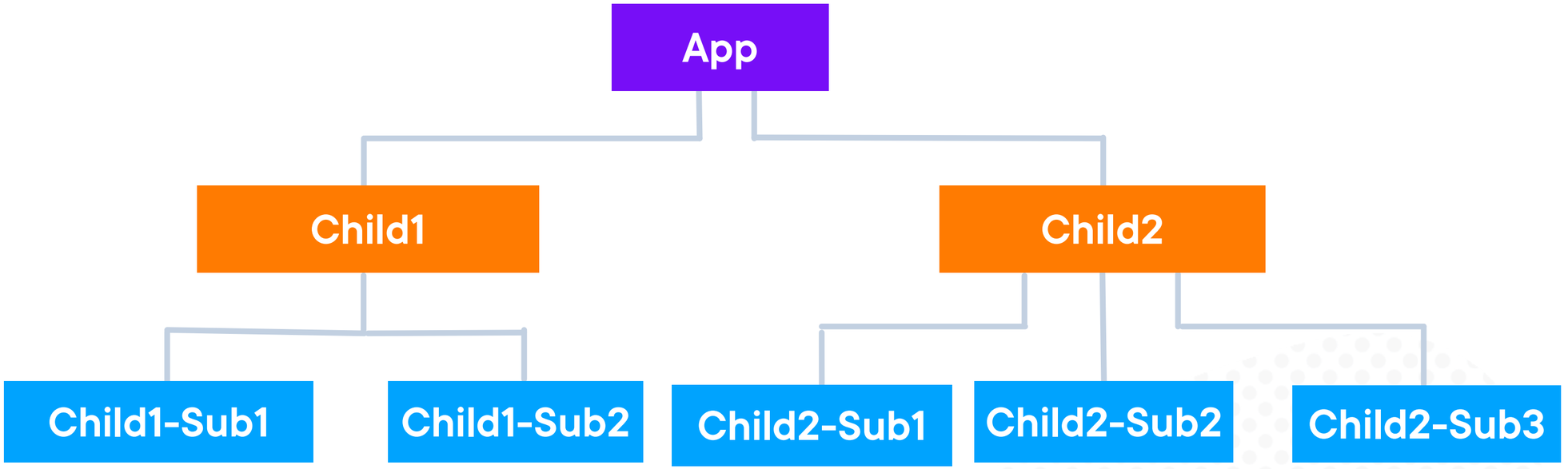


React Component Hierarchy

React Component Hierarchy

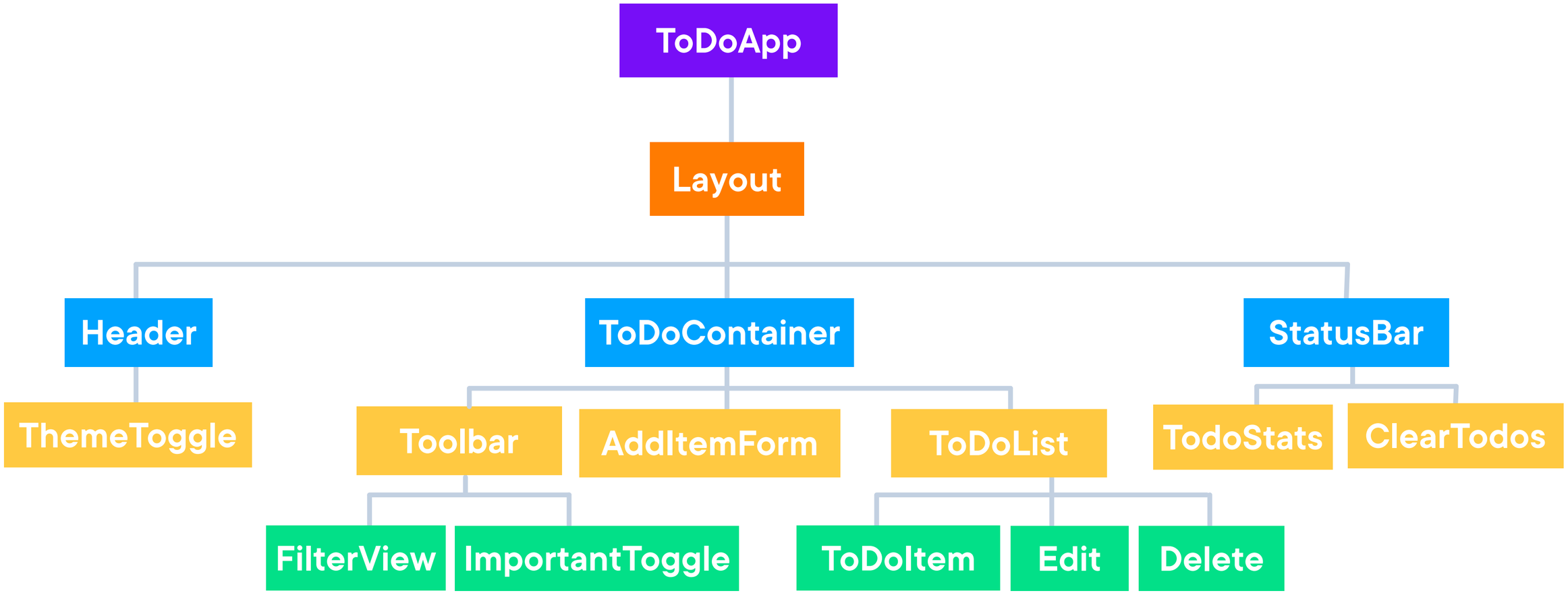


React Component Hierarchy

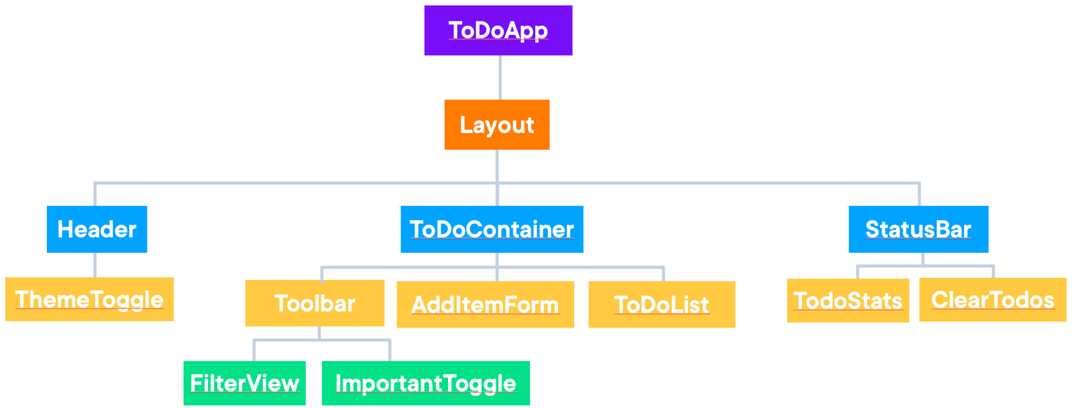


React Component Hierarchy

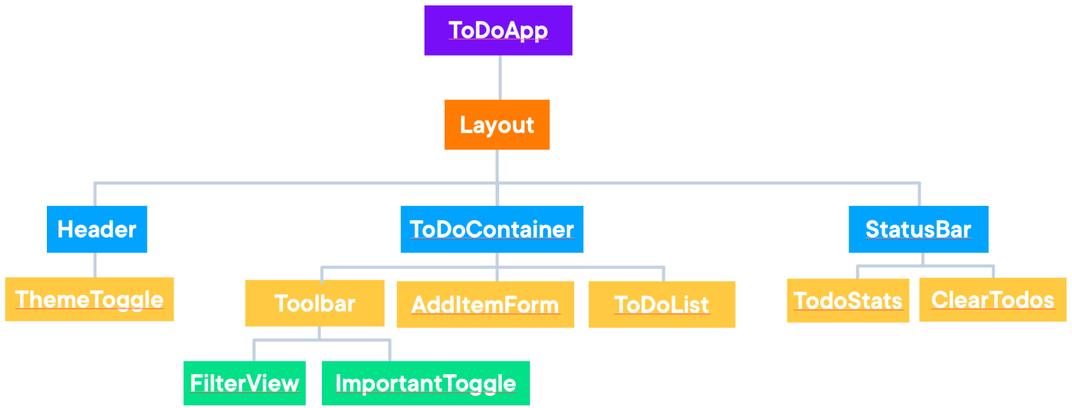
React Component Hierarchy

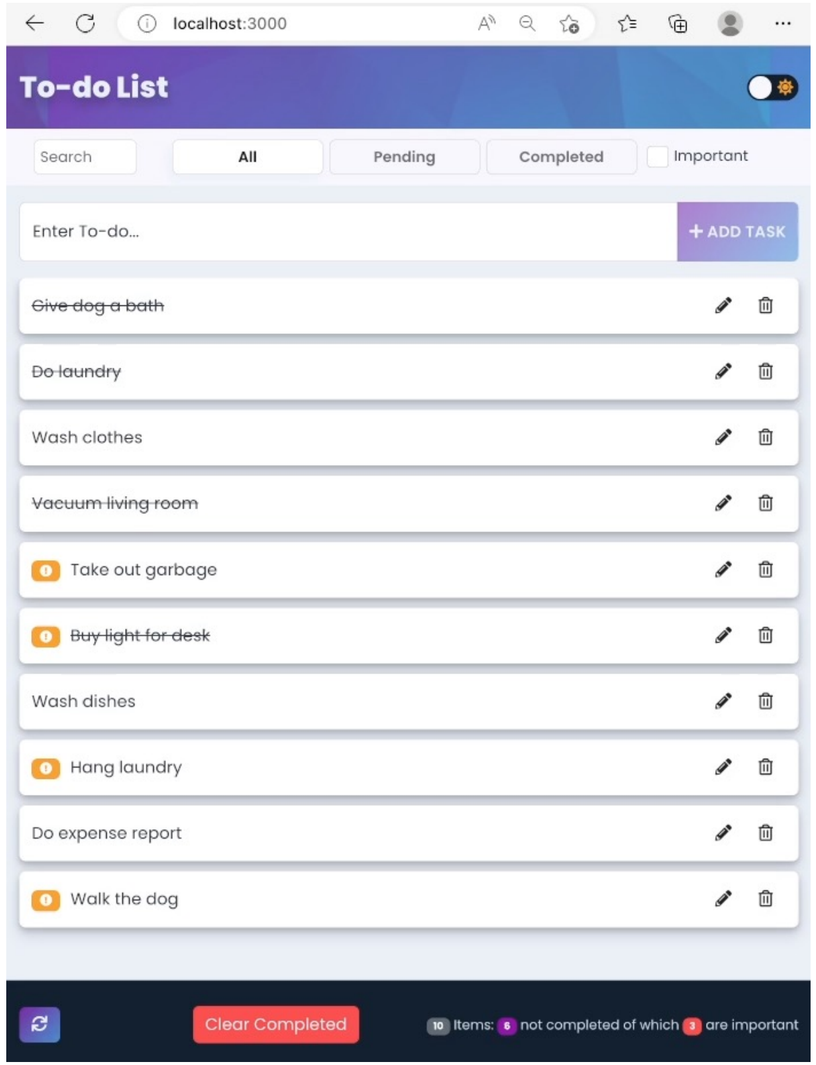


React Component Hierarchy



React Component Hierarchy





A Deep Dive into a Single Component

The ToDoItem Component

ToDoItem.js

The ToDoItem Component

ToDoItem.js

```
function ToDoItem() {
  return (
    <div></div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem() {
  const text = "Wash clothes";
  return (
    <div></div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem() {
  const text = "Wash clothes";
  return (
    <div>{text}</div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem() {
  const text = "Wash clothes";
  const completed = false;
  return (
    <div>{text}</div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem() {
  const text = "Wash clothes";
  const completed = false;
  return (
    <div className={completed ? "completed" : ""} >{text}</div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem() {
  const text = "Wash clothes";
  const completed = false;
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem(props) {
  const text = "Wash clothes";
  const completed = false;
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem(props) {
  const props.text = "Wash clothes";
  const props.completed = false;
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem(props) {
  const { text, completed }  = props;

  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed }) {


  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed }) {
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed }) {
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

Techniques for Passing Props

The ToDoItem Component

ToDoItem.js

```
function ToDoItem(props) {
  const { text, completed }  = props;

  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed }) {


  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed }) {
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed, important }) {
  return (
    <div className={completed ? "completed" : ""} >
      {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed, important }) {
  return (
    <div className={completed ? "completed" : ""} >
                                {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed, important }) {
  return (
    <div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  return <div></div>;
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  return <ToDoItem />;
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  return (<ToDoItem text="Wash clothes" completed={false}
                    important={true} />);
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
       {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  return (<ToDoItem text="Wash clothes" completed={false}
                                        />);
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
       {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  return (<ToDoItem text="Wash clothes" completed={false}
                    important={true} />);
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
       {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  const toDoItem =
    { text: "Wash clothes", completed: false, important: true, }
  return (<ToDoItem />);
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  const toDoItem =
    { text: "Wash clothes", completed: false, important: true, }
  return (<ToDoItem {...toDoItem} />);
}
```

```
function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  const toDoItem =
    { text: "Wash clothes", completed: false, important: true,
      secretData: "MySecretData",}
  return (<ToDoItem {...toDoItem} />);
}



function ToDoItem({ text, completed, important }) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  const toDoItem =
    { text: "Wash clothes", completed: false, important: true,
      secretData: "MySecretData",}
  return (<ToDoItem {...toDoItem} />);
}



function ToDoItem(props) {
  return (<div className={completed ? "completed" : ""} >
      {important ? "*" : "" } {text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  const toDoItem =
    { text: "Wash clothes", completed: false, important: true,
      secretData: "MySecretData",}
  return (<ToDoItem {...toDoItem} />);
}



function ToDoItem(props) {
  return (<div className={props.completed ? "completed" : ""} >
      {props.important ? "*" : "" } {props.text}</div>);
}
```

The ToDoItem Component

ToDoItem.js

```
export default function Index() {
  const toDoItem =
    { text: "Wash clothes", completed: false, important: true,
      secretData: "MySecretData",}
  return (<ToDoItem {...toDoItem} />);
}



function ToDoItem(props) {
  return (<div className={props.completed ? "completed" : ""} >
      {props.important ? "*" : "" } {props.text}</div>);
      { JSON.stringify(props) }       }
```

React is more than about rendering static HTML

The reason React is called React is because components need to change what they render

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

```
<html lang="en">
  <body>
    <script>

      document.addEventListener
      ("DOMContentLoaded", () => {
        console.log("Ready!");
        });

    </script>
  </body>
</html>
```

```
function ToDoItem({ text, completed, important }) {

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {text}
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  useState(text);
  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {text}
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {text}
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);

  useEffect();

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);

  useEffect(() => {}, []);

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);

  useEffect(() => {
    setTextState();
  }, []);

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);

  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div className={completedState ? "completed" : ""}>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div className={completedState ? "completed" : ""}
         onClick={ () => {} }>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div className={completedState ? "completed" : ""}
         onClick={ () => {
                     setCompletedState(!completedState);
                     } }>
      {important ? "*" : ""} {textState }
    </div>
  );
}
```

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

```
function ToDoItem({ text, completed, important }) {

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {text}
    </div>
  );
}
```

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

```
function ToDoItem({ text, completed, important }) {

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {text}
    </div>
  );
}
```

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);

  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState}
    </div>
  );
}
```

Basic component is timeless

Component with side effects, after rendering

Component with events, buttons

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);

  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div onClick={() => {
                        setCompletedState(!completedState);
                    }}>
      <div className={completedState ? "completed" : ""}>
        {important ? "*" : ""} {textState}
      </div>
    </div>
  );
}
```

Course Expectations

Components are building blocks for real world React apps

Components are just associated JavaScript functions

Components are built using helper functions in the React library

Course Expectations

Components are building blocks for real world React apps

Components are just associated JavaScript functions

Components are built using helper functions in the React library

Course Expectations

Components are building blocks for real world React apps

Components are just associated JavaScript functions

Components are built using helper functions in the React library

Course Expectations

Components are building blocks for real world React apps

Components are just associated JavaScript functions

Components are built using helper functions in the React library

Course Intention

Improve working with components

Make React apps easier to build and maintain

Provide better user experiences

Course Intention

Improve working with components

Make React apps easier to build and maintain

Provide better user experiences

Course Intention

Improve working with components

Make React apps easier to build and maintain

Provide better user experiences

Course Intention

Improve working with components

Make React apps easier to build and maintain

Provide better user experiences

Coming Up in This Course

Exploring a React app

Global state and context

Handling errors and debugging

Improving performance

Server components

Coming Up in This Course

Exploring a React app

Global state and context

Handling errors and debugging

Improving performance

Server components

Coming Up in This Course

Exploring a React app

Global state and context

Handling errors and debugging

Improving performance

Server components

Coming Up in This Course

Exploring a React app

Global state and context

Handling errors and debugging

Improving performance

Server components

Coming Up in This Course

Exploring a React app

Global state and context

Handling errors and debugging

Improving performance

Server components

Coming Up in This Course

Exploring a React app

Global state and context

Handling errors and debugging

Improving performance

Server components

END OF SLIDES

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div>
      <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
      </div>
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div onClick={ () => {} }>
      <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
      </div>
    </div>
  );
}
```

```
function ToDoItem({ text, completed, important }) {
  const [textState, setTextState] = useState(text);
  const [completedState, setCompletedState] =
     useState(completed);


  useEffect(() => {
    setTextState
       (`${text} ${new Date().toLocaleTimeString()}`);
  }, []);

  return (
    <div onClick={() => {
                        setCompletedState(!completedState);
                    }}>
      <div className={completed ? "completed" : ""}>
      {important ? "*" : ""} {textState }
      </div>
    </div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem(props) {
  return (
    <div className={props.completed ? "completed" : ""} >
      {props.important ? "*" : "" } {props.text}</div>
  );
}
```

The ToDoItem Component

ToDoItem.js

```
function ToDoItem({ text, completed, important }) {

  const [completedState, setCompletedState] = useState(completed);

  return (
    <div className={completedState ? "completed" : ""}
         onClick={() => {
           setCompletedState(!completedState);
         }}>
      {important ? "*" : "" } {text}
    </div>
  );
}
```

Timeline of Events with Fade Transition

Time 0

Parameters passed in and component renders

Time 1

After component full renders

Time 2

Click event fires surrounding text

Parameters:
text: “Wash clothes”
completed: false

Stephen Cole Kleene

1973

Ken Thompson
First release of *grep*

1987

Larry Wall
Integration into Perl

Timeline of Events with Fade Transition

1940s
McCulloch and Pitts
Neural network theory

1956
Stephen Cole Kleene
Regular events/sets

1968
Ken Thompson
First in computing

1973
Ken Thompson
First release of *grep*

1986
Henry Spencer
Regex library

1987
Larry Wall
Integration into Perl

*View in presentation mode

Timeline of Events with Fade Transition

1940s
McCulloch and Pitts
Neural network theory

1956
Stephen Cole Kleene
Regular events/sets

1968
Ken Thompson
First in computing

1973
Ken Thompson
First release of *grep*

1986
Henry Spencer
Regex library

1987
Larry Wall
Integration into Perl

*View in presentation mode

The ToDoItem Component

Filename.code

```
<div class="row carousel-indicators">
    <div style="background-color:red;" class="col-md-4" data-
target="#homeCarousel" data-slide-to="0" class="active">
    </div>
    <div style="background-color:green;" class="col-md-4" data-
target="#homeCarousel" data-slide-to="1" class="active">
    </div>
</div>
```