Building Front-End Applications, Rapidly

Parish & Carlos

iCIMS Labs

icims®

Agenda

- What is React
- Intro to JSX
- Hands on, create a simple component
- 3rd Party Components
- Intro to create-react-app
- Hands on, create a Todo App
- Hands on, test Todo App

Prerecquisits

- Knowledge of Javascript, HTML & CSS
- Node.js 17 @ nodejs.org
- Pref: Visual Studio Code @ code.visualstudio.com

Follow Along

- https://github.com/Disco-Pope/rapid-react-todo
- https://rapid-react-app.firebaseapp.com/#0

Objectives and Expectations

Upon completing this workshop, you will:

- Have an understanding of React
- How to create simple React components
- How to use 3rd party React components
- Build a simple ToDo app using `create-react-app`

ReactJS

What is React?

- A Javascript library for building user interfaces created by Facebook
- The V in M**V**C
- Component based (components on components on components!)
- Key terms: Components, Props, State

Virtual DOM

- 🐢DOM is slow (re-rendering slows you down)
- ⚡Virtual DOM does no rendering
- React ONLY updates objects that have changed in the real DOM
- DOM updates are batched

JSX

- Javascript + XML
- Syntax extension to JavaScript
- HTML with some modifications
  - `className` instead of `class`
  - `children` prop
- Easier to read & write

Using Javascript

```
1. function Hello(props) {
2.   return React.createElement(
3.     'div',
4.     { style: { fontSize: '20px' }, className: 'wrapper' },
5.     `Hello ${props.toWho}!`
6.   );
7. }
8.
9. ReactDOM.render(
10.   React.createElement(Hello, {toWho: 'World'}, null),
11.   document.getElementById('root')
12. ),
```

With JSX

```
1. function Hello(props) {
2.   return (
3.     <div
4.       className="wrapper"
5.       style={{fontSize: '20px'}}
6.     >
7.       Hello {props.toWho}!
8.     </div>
9.   );
10. }
11.
12. ReactDOM.render(
```

react clock - npm search

npmjs.com/search?q=react%20clock

Narcissistic Passion Minified

npm Enterprise Products Solutions Resources Docs Support

npm react clock Search Join Log In

90 packages found

1 2 3 ... 5 »

Sort Packages

Optimal

Popularity

Quality

Maintenance

Who's Hiring?

react-clock

An analog clock for your React app.

clock digital clock analog clock time react

wojtekmaj published 2.4.0 • 2 months ago

@meecolabs/react-clock

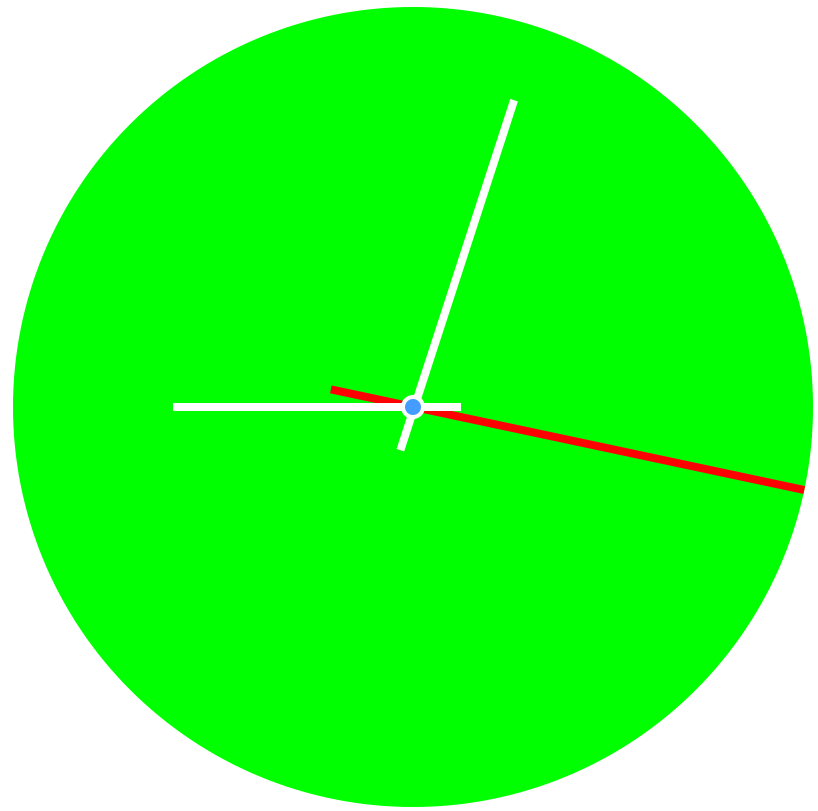
An analog clock for your React app.

Component Props

Props

Prop name	Description	Example values
className	Defines class name(s) that will be added along with "react-clock" to the main React-Clock `<time>` element.	• String: `"class1 class2"` • Array of strings: `["class1", "class2 class3"]`
hourHandLength	Defines the length of an hour hand, in %. Defaults to `50`.	`80`
hourHandOppositeLength	Defines the length of the part of an hour hand on the opposite side the hand is pointing to, in %. Defaults to `10`.	`20`
hourHandWidth	Defines the width of an hour hand, in pixels. Defaults to `4`.	`3`
hourMarksLength	Defines the length of hour marks, in %. Defaults to `10`.	`8`
hourMarksWidth	Defines the width of hour marks, in pixels. Defaults to `3`.	`2`
minuteHandLength	Defines the length of a minute hand, in %. Defaults to `70`.	`80`
minuteHandOppositeLength	Defines the length of the part of a minute hand on the opposite side the hand is pointing to, in %. Defaults to `10`.	`20`
minuteHandWidth	Defines the width of a minute hand, in pixels. Defaults to `2`.	`3`
minuteMarksLength	Defines the length of minute marks, in %. Defaults to `6`.	`8`
minuteMarksWidth	Defines the width of a minute hand, in pixels. Defaults to `1`.	`2`
renderHourMarks	Defines whether hour marks shall be rendered. Defaults to `true`.	`false`
[illegible]	Defines whether minute hand shall be rendered.	[illegible]

Analog Clock



Let's build a todo list!

Let's build a todo list!

1. Create a react project
2. Adding a basic list
3. Add state and controls
4. Make our todo list responsive
5. Ways to take your app further

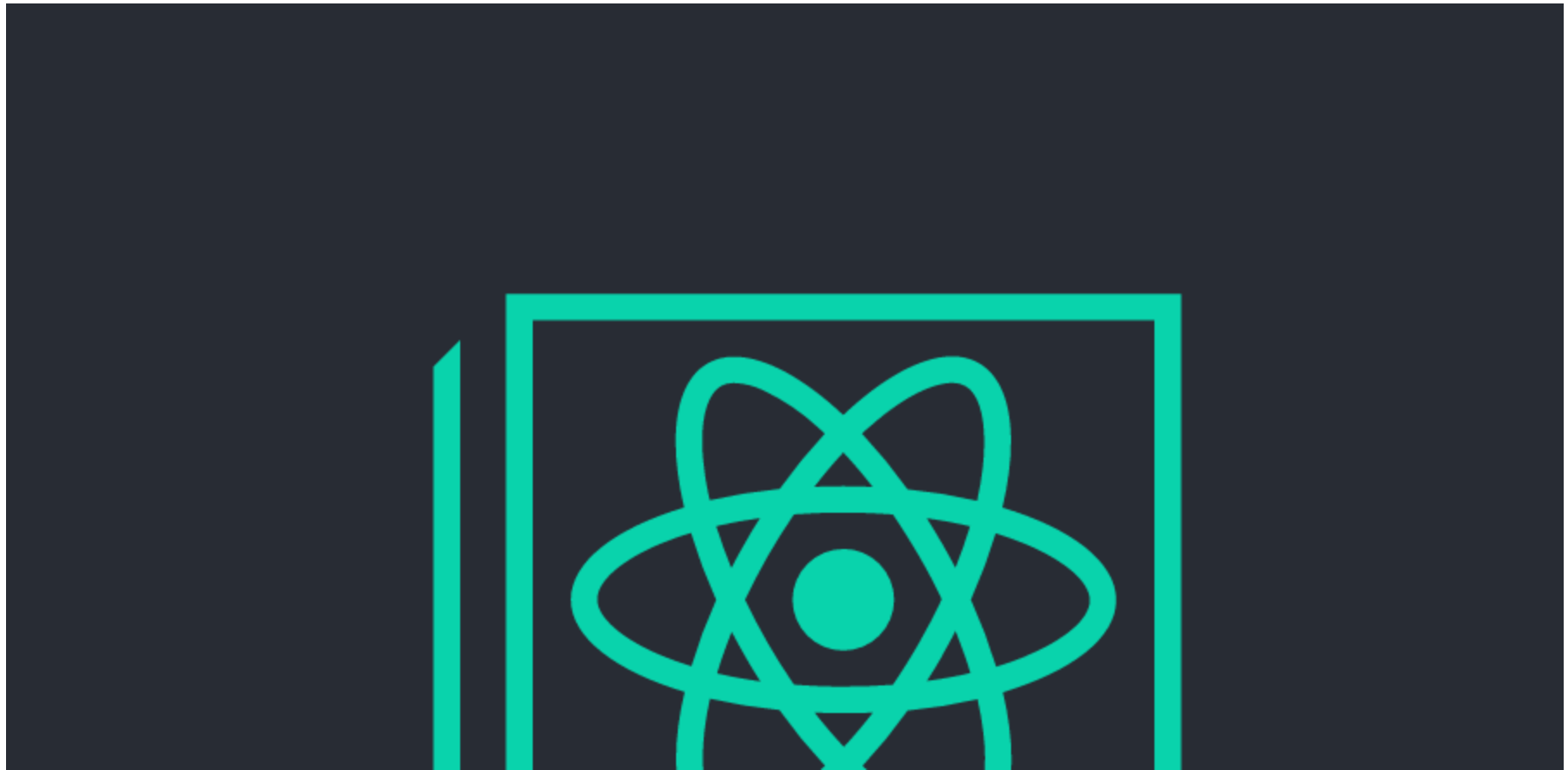
create-react-app

- Written by facebook for learning react
- Generates a full react project with a single command
- Serious time saver
- Lots of tooling is configured for you
- Hot-reloading dev server

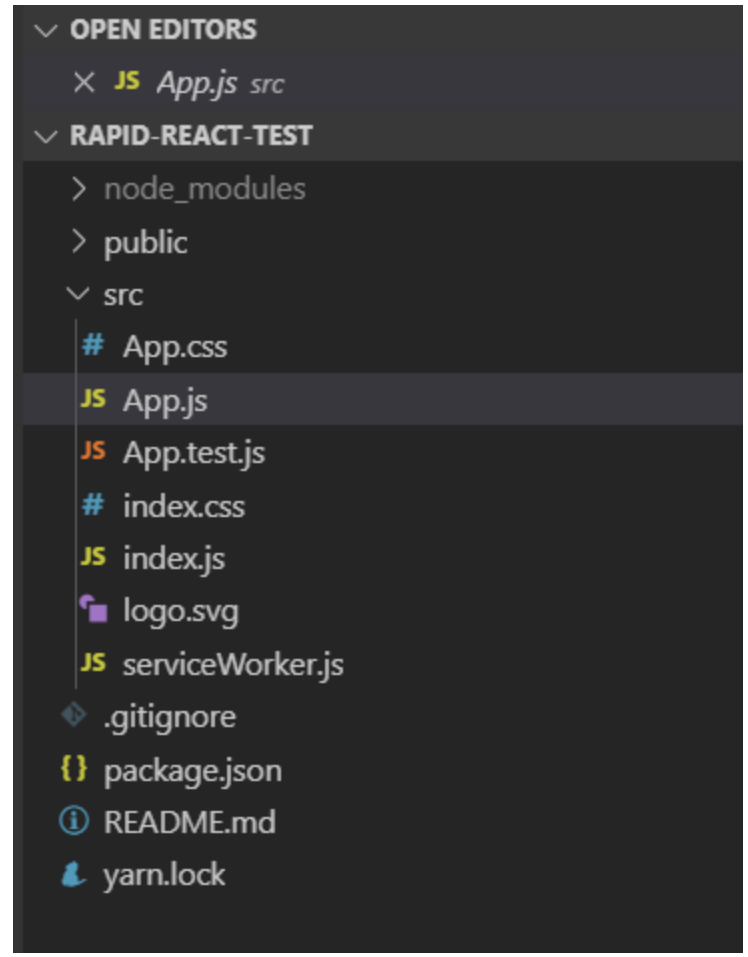
Generating a project

```
npx create-react-app rapid-react-todo
```

- Open in VS Code
- Open terminal (ctrl + `)
- Execute the dev server with `npm start`



The project structure will look like this



open app.js

src/App.js

Lets clean out the app so we can write our todo list code here

Remove lines 2-3

```
1. import React from 'react';
2.
3. function App() {
4.   return (
5.     <>
6.       <p>test</p>
7.     </>
8.   );
9. }
10.
11. export default App;
```

Your code will look like this

Your app will look like this

How your app looks in HTML

```
▼<div id="root">
    <p>test</p>
  </div>
```

All of your app renders into the "root" element

Component Libraries

- Default HTML Elements look bad

- Great looking components
- Features HTML doesn't have
- Components HTML doesn't have

Material UI

- Check out Material UI
- React component library based on Material Design
- Install Material UI Core Components and Icons
  - `npm install --save @material-ui/core @material-ui/icons`

Material Components for Todo List

- List, ListItem, etc (Examples | Docs)
- Button, IconButton (Examples | Docs)
- TextField (Examples | Docs)
- DeleteIcon (Examples | Docs)
- Grid (Examples | Docs)

Component Example Pages

- Lots of examples of components
- Live editing of all examples

```
      <label htmlFor="contained-button-file">
        <Button variant="contained" component="span" className={classes.button}>
          Upload
        </Button>
      </label>
    </div>
  );
}
```

DEFAULT PRIMARY SECONDARY DISABLED LINK UPLOAD

Button API

The API documentation of the Button React component. Learn more about the props and the CSS customization point

Import

```
import Button from '@material-ui/core/Button';
// or
```

```
1. import React from 'react';
2. import { List, ListItem, ListItemText } from '@material-ui/core';
3.
4. function App() {
5.   return (
6.     <>
7.       <p>test</p>
8.     </>
9.   );
10. }
11.
12. export default App;
```

Let's start by adding some todos to our page

```
1. import React from 'react';
2. import { List, ListItem, ListItemText } from '@material-ui/core';
3.
4. function App() {
5.   return (
6.     <>
7.       <List>
8.         <ListItem>
9.           <ListItemText>Todo1</ListItemText>
10.        </ListItem>
11.        <ListItem>
12.          <ListItemText>Todo2</ListItemText>
13.        </ListItem>
14.      </List>
15.    </>
```

We've added a static list with 2 todo items

Your app with list items

test1

test2

About React Hooks and State

- React Hooks are functions that hook extra behavior into our components
- The useState hook lets our component remember things between renders
- Let's add state to our todo list

We need to import "useState" from react like this

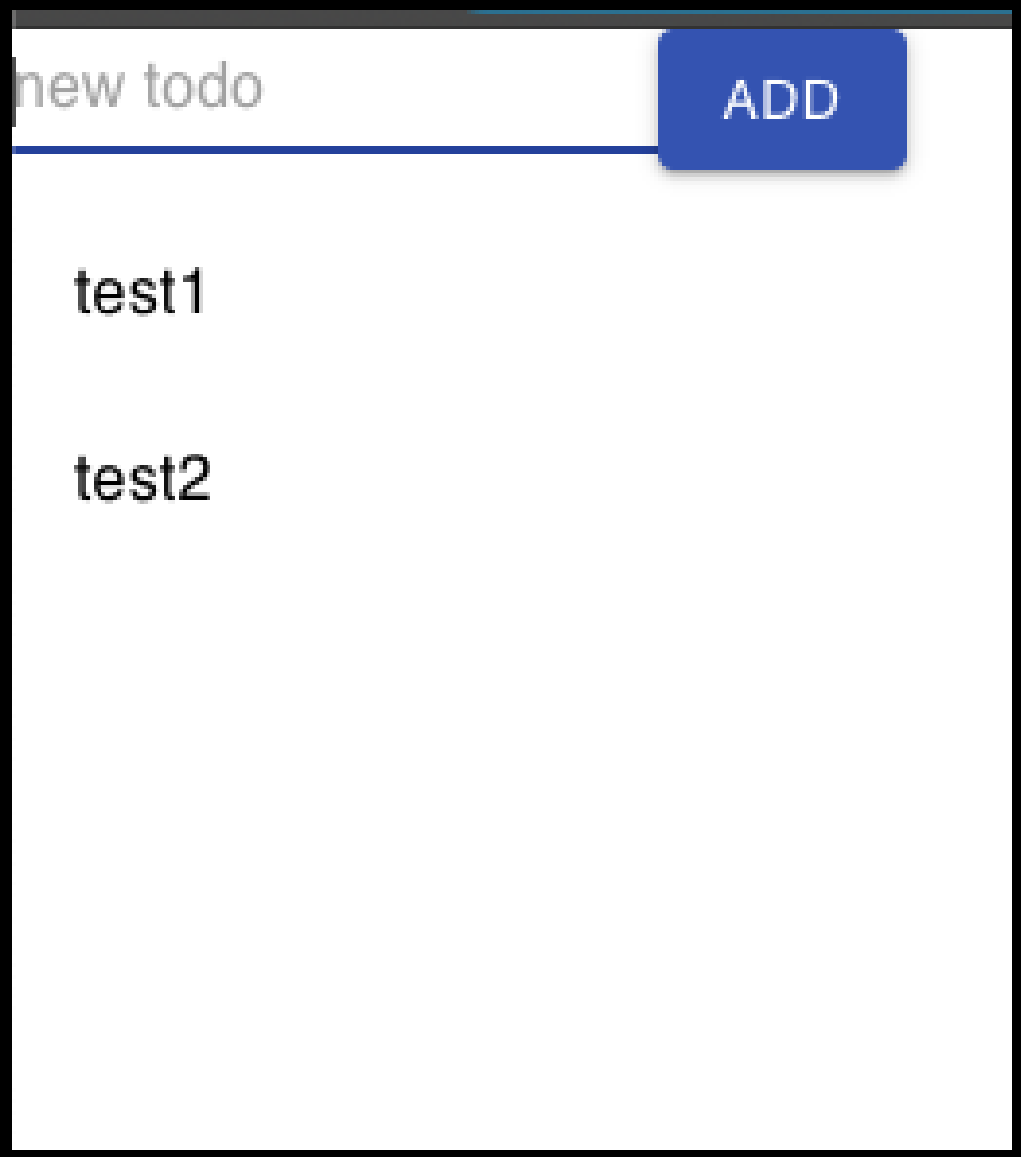
Adding new todos

- Our todos are part of state but our users can't manipulate them
- Lets add controls for adding new todos to our state
- A React "Ref" is used to access the HTML element rendered by react, we use this to read the value a user enters into a text field

```
1. import React, { useState, useRef } from 'react';
2. import {
3.   Button,
4.   List,
5.   ListItem,
6.   ListItemText,
7.   TextField
8. } from '@material-ui/core';
9.
0. function App() {
1.   const textFieldRef = useRef(null);
2.   const [todos, setTodos] = useState(['test1', 'test2']);
3.
4.   // add new todo to state and clear the input
5.   function handleAddClick() {
6.     const newTodo = textFieldRef.current.value;
7.     if(newTodo.length > 0) {
8.       setTodos([...todos, newTodo]);
```

Import useRef from react

Your app with add controls



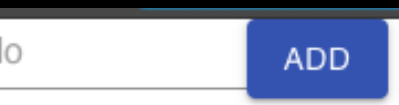
Removing todos

- Lets add a trash button for removing todos
- Being able to remove todos is useful for marking things completed

```
1. import React, { useState, useRef } from 'react';
2. import {
3.   Button,
4.   IconButton,
5.   List,
6.   ListItem,
7.   ListItemText,
8.   ListItemSecondaryAction,
9.   TextField
0. } from '@material-ui/core';
1. import { Delete as DeleteIcon } from '@material-ui/icons';
2.
3. function App() {
4.   const textFieldRef = useRef(null);
5.   const [todos, setTodos] = useState(['test1', 'test2']);
6.
7.   // add new todo to state and clear the input
8.   function handleAddClick() {
9.     const newTodo = textFieldRef.current.value;
```

Import IconButton and ListItemSecondaryAction

Your app with remove controls



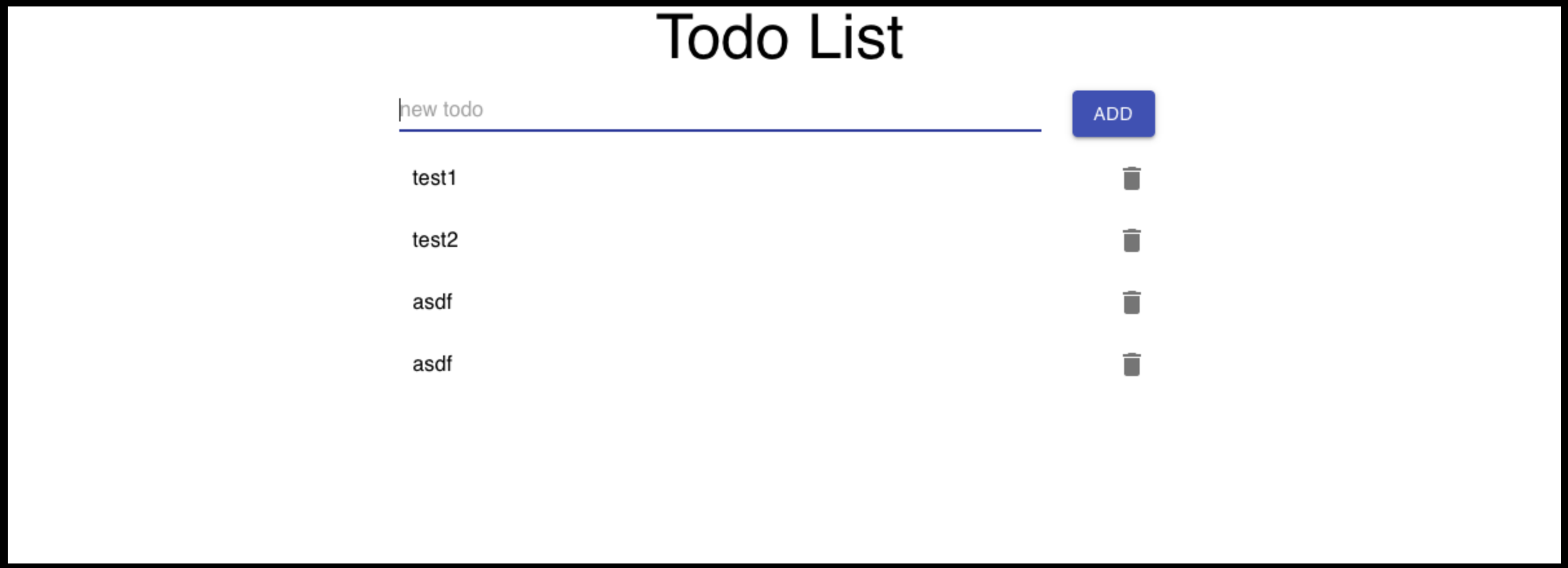
Responsive Layout

- Let's add a responsive Layout
- We'll use Material-UIs Grid component to make things look much better

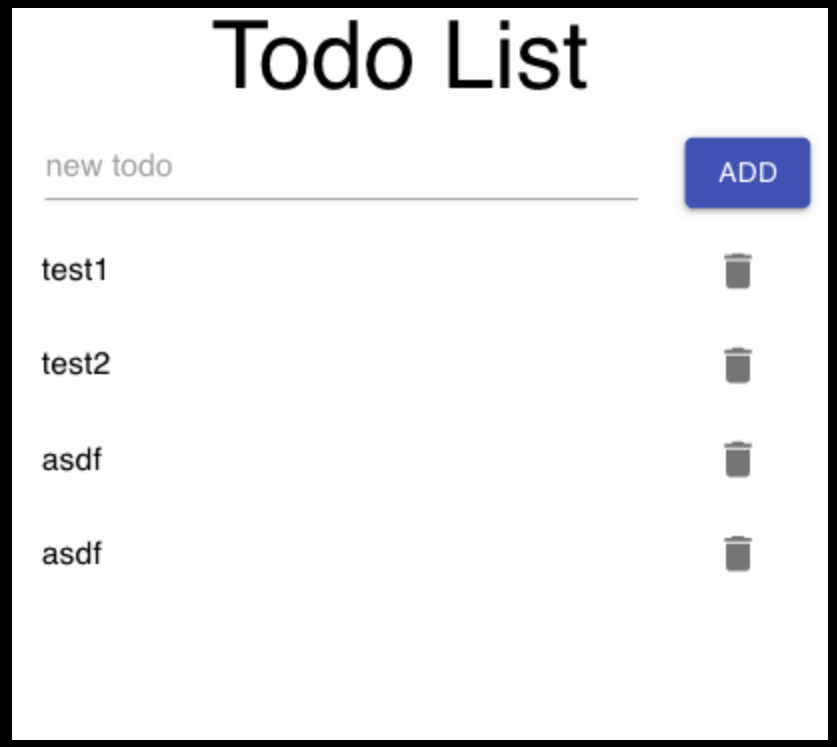
```
import React, { useState, useRef } from 'react';
import {
  Button,
  Grid,
  IconButton,
  List,
  ListItem,
  ListItemText,
  ListItemSecondaryAction,
  TextField,
  Typography
} from '@material-ui/core';
import { Delete as DeleteIcon } from '@material-ui/icons';

function App() {
  const textFieldRef = useRef(null);
  const [todos, setTodos] = useState(['test1', 'test2']);

  // add new todo to state and clear the input
```

Import Grid and Typography

Desktop



Phone



Break it down

```
 1. import React, { useState, useRef } from 'react';
 2. import {
 3.   Button,
 4.   Grid,
 5.   IconButton,
 6.   List,
 7.   ListItem,
 8.   ListItemText,
 9.   ListItemSecondaryAction,
10.   TextField,
11.   Typography
12. } from '@material-ui/core';
13. import { Delete as DeleteIcon } from '@material-ui/icons';
14.
15. function App() {
16.   const textFieldRef = useRef(null);
17.   const [todos, setTodos] = useState(['test1', 'test2']);
18.
19.   // add new todo to state and clear the input
20.   function handleAddClick() {
21.     const newTodo = textFieldRef.current.value;
22.     if(newTodo.length > 0) {
23.       setTodos([...todos, newTodo]);
24.       textFieldRef.current.value = '';
25.     }
26.   };
27.
28.   // update state with deleted todo filtered out
29.   function handleDeleteClick(index) {
30.     setTodos(oldTodos => oldTodos.filter((e, i) => i !== index));
31.   }
```

lib/Form.js

```
 1. import React, { useState, useRef } from 'react';
 2. import {
 3.   // Button,
 4.   Grid,
 5.   IconButton,
 6.   List,
 7.   ListItem,
 8.   ListItemText,
 9.   ListItemSecondaryAction,
10.   // TextField,
11.   Typography
12. } from '@material-ui/core';
13. import { Delete as DeleteIcon } from '@material-ui/icons';
14.
15. import Form from './lib/Form';
16.
17. function App() {
18.   const textFieldRef = useRef(null);
19.   const [todos, setTodos] = useState(['test1', 'test2']);
20.
21.   // add new todo to state and clear the input
22.   function handleAddClick() {
23.     const newTodo = textFieldRef.current.value;
24.     if(newTodo.length > 0) {
25.       setTodos([...todos, newTodo]);
26.       textFieldRef.current.value = '';
27.     }
28.   };
29.
30.   // update state with deleted todo filtered out
```

```
 1. import React, { useState, useRef } from 'react';
 2. import {
 3.   Grid,
 4.   IconButton,
 5.   List,
 6.   ListItem,
 7.   ListItemText,
 8.   ListItemSecondaryAction,
 9.   Typography
10. } from '@material-ui/core';
11. import { Delete as DeleteIcon } from '@material-ui/icons';
12.
13. import Form from './lib/Form';
14.
15. function App() {
16.   const textFieldRef = useRef(null);
17.   const [todos, setTodos] = useState(['test1', 'test2']);
18.
19.   // add new todo to state and clear the input
20.   function handleAddClick() {
21.     const newTodo = textFieldRef.current.value;
22.     if(newTodo.length > 0) {
23.       setTodos([...todos, newTodo]);
24.       textFieldRef.current.value = '';
25.     }
26.   };
27.
28.   // update state with deleted todo filtered out
29.   function handleDeleteClick(index) {
```

lib/TodoItem.js

```
 1. import React, { useState, useRef } from 'react';
 2. import {
 3.   Grid,
 4.   Typography
 5.  } from '@material-ui/core';
 6.
 7. import Form from './lib/Form';
 8. import TodoItem from './lib/TodoItem';
 9.
10. function App() {
11.   const textFieldRef = useRef(null);
12.   const [todos, setTodos] = useState(['test1', 'test2']);
13.
14.   // add new todo to state and clear the input
15.   function handleAddClick() {
16.     const newTodo = textFieldRef.current.value;
17.     if(newTodo.length > 0) {
18.       setTodos([...todos, newTodo]);
19.       textFieldRef.current.value = '';
20.     }
21.   };
22.
23.   // update state with deleted todo filtered out
24.   function handleDeleteClick(index) {
25.     setTodos(oldTodos => oldTodos.filter((e, i) => i !== index));
26.   }
27.
28.   return (
```

Automated Testing

- create-react-app comes with jest testing built in
- jest will treat files with .test.js extension as test files
- create-react-app includes a very simple test for App.js
- run jest using `npm test`
  - automatically starts in watch mode

Jest Watch mode

```
PASS  src/App.test.js
  ✓ renders without crashing (90ms)

Test Suites: 1 passed, 1 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        1.941s, estimated 2s
Ran all test suites.

Watch Usage: Press w to show more.
```

Benefits of Automated Tests

- As your app gets bigger it becomes harder to ensure nothing breaks as you make changes
- If your tests are all written in code you can run all of them in moments and have confidence that your changes didn't have unforeseen consequences

App.test.js

```
1. import React from 'react';
2. import ReactDOM from 'react-dom';
3. import App from './App';
4.
5. it('renders without crashing', () => {
6.   const div = document.createElement('div');
7.   ReactDOM.render(<App />, div);
8.   ReactDOM.unmountComponentAtNode(div);
9. });
```

Quick anatomy of tests

Enzyme

- A react testing library by Airbnb
- Enzyme can "shallow render" components
  - Renders your component without rendering all its chilren
  - Very fast and is sufficient for many tests

Let's write a test for Form.js

- We'll need to install enzyme
  - `npm install --save-dev enzyme enzyme-adapter-react-16`

lib/Form.test.js

```
1. import React from 'react';
2. import { Button } from '@material-ui/core';
3. import Form from './Form';
4.
5. import { shallow, configure } from 'enzyme';
6. import Adapter from 'enzyme-adapter-react-16';
7. configure({ adapter: new Adapter() });
8.
9. it('should invoke onAddClick when add button is clicked', () => {
10.     const mockAddFn = jest.fn();
11.     const wrapper = shallow(<Form onAddClick={mockAddFn} />);
12.
```

Import react and the Button component from material ui

```
PASS  src/App.test.js
PASS  src/lib/Form.test.js

Test Suites: 2 passed, 2 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        3.281s
Ran all test suites.

Watch Usage: Press w to show more.
```

Ideas for taking your app further

- Ability to edit todos
  - Store additional state for edit mode
  - Use text field instead of ListItemText
- Saving todos to a local storage
  - save on add, remove, edit
- Syncing todos to backend
  - Trigger backend sync after any change to local storage
  - Use mocks to develop without a real backend
    - axios, axios-mock-adapter
- Create a backend with expressjs

Q & A

Thank you!