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
- Hands on, create-react-app
- Hands on, using a 3rd party component (Material UI)

Prerecquisits: Javascript, HTML, CSS & Node.js

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
- The V in M**V**C
- VirtualDOM
- Component based (components on components on components!)
- Key terms: Components, Props, State

JSX

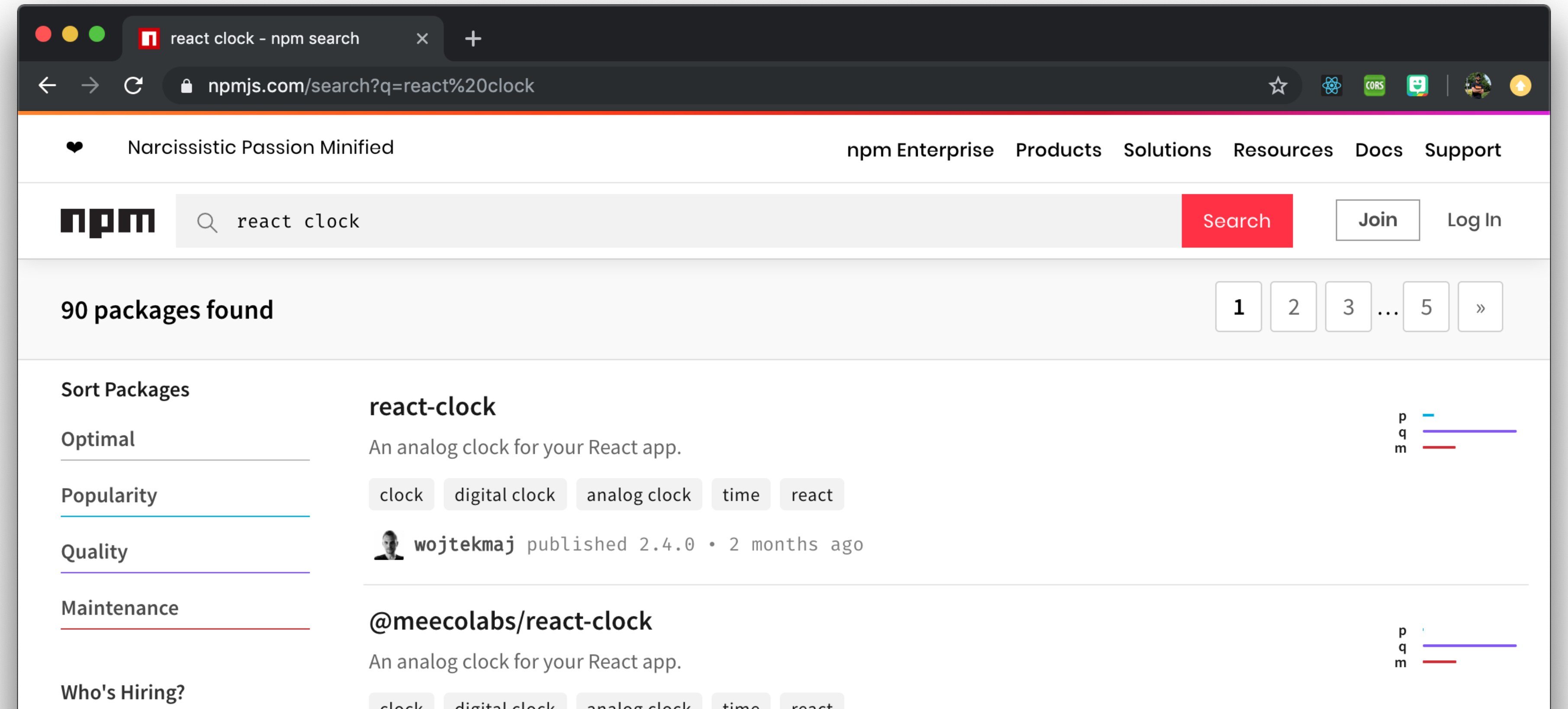
- Javascript + XML
- Extension of ES6
- HTML with some modifications
  - `className` instead of `class`
  - `children` prop
- Easier to read & write

Using Javascript

```javascript
function Hello(props) {
  return React.createElement(
    'div',
    { style: { fontSize: '20px' }, className: 'wrapper' },
    `Hello ${props.toWho}!`
  );
}

ReactDOM.render(
  React.createElement(Hello, {toWho: 'World'}, null),
  document.getElementById('root')
);
```

With JSX

```jsx
function Hello(props) {
  return (
    <div
      className="wrapper"
      style={{fontSize: '20px'}}
    >
      Hello {props.toWho}!
    </div>
  );
}

ReactDOM.render(
```


react clock - npm search

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

❤ Narcissistic Passion Minified

npm Enterprise Products Solutions Resources Docs Support

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.

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

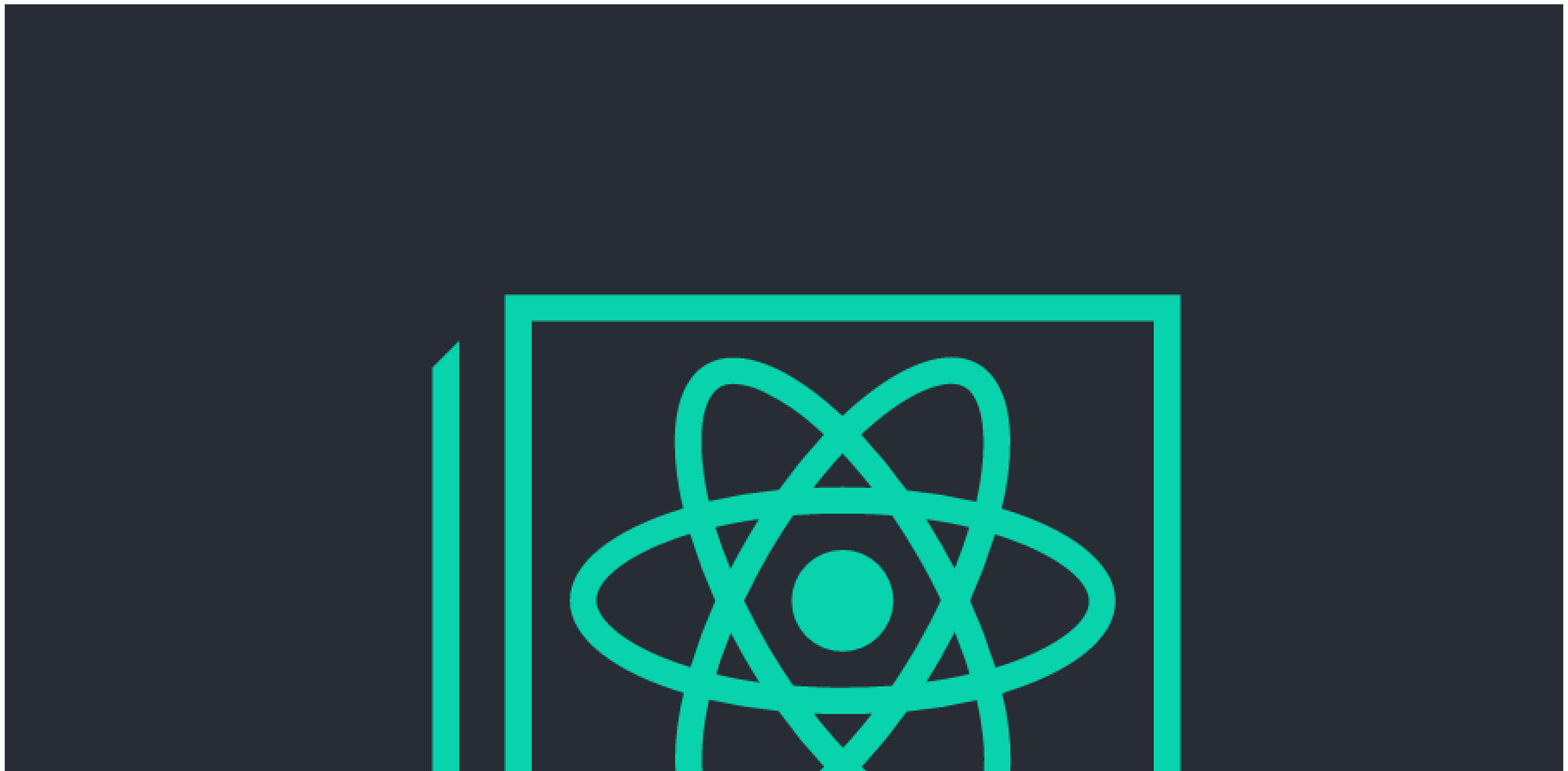
create-react-app

- 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

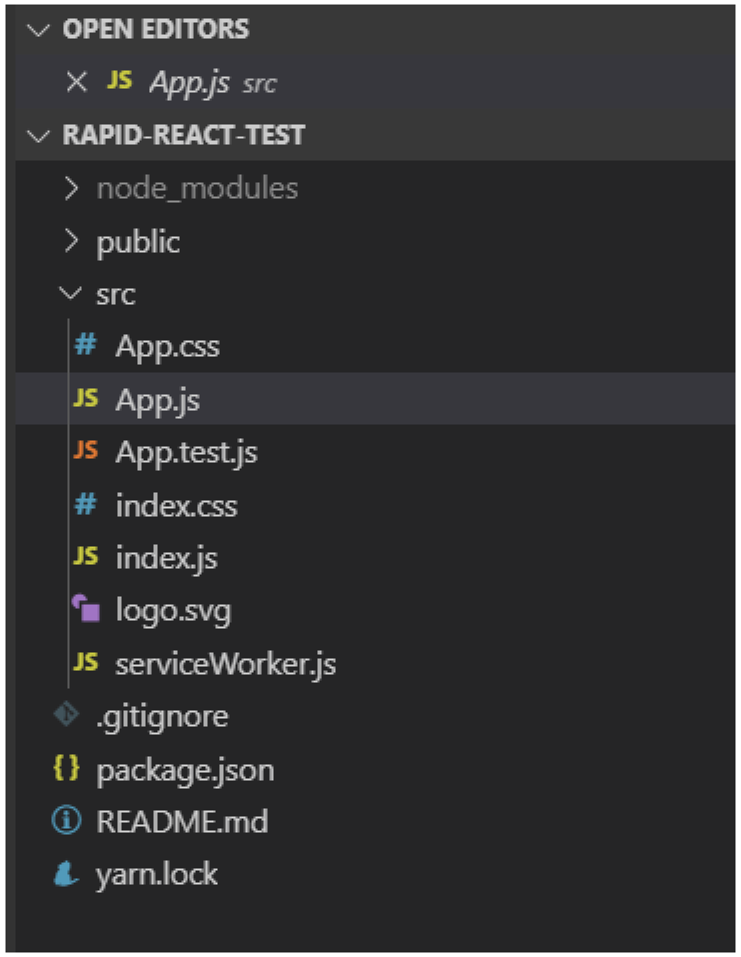
Generating a project

```
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

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

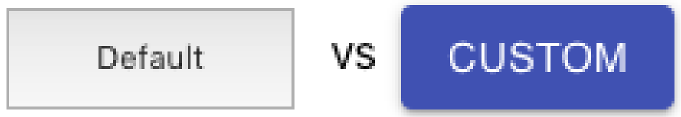
Remove lines 2-3

```jsx
import React from 'react';

function App() {
  return (
    <>
      <p>test</p>
    </>
  );
}

export default App;
```

Your code will look like this

Component Libraries

- Default HTML Elements look bad



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

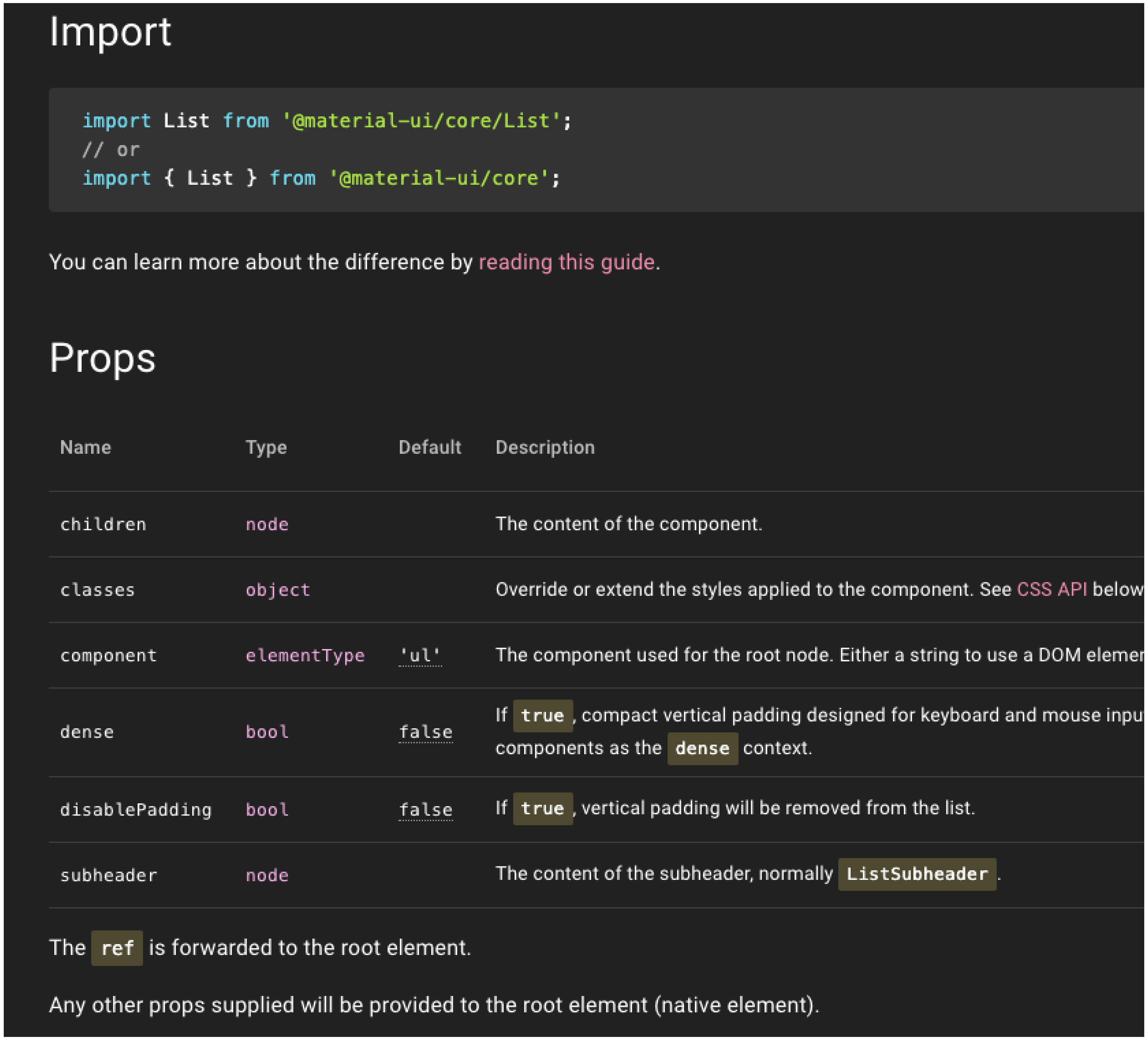
Material UI

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



Import

```
import List from '@material-ui/core/List';
// or
import { List } from '@material-ui/core';
```

You can learn more about the difference by reading this guide.

Props

Name	Type	Default	Description
children	node		The content of the component.
classes	object		Override or extend the styles applied to the component. See CSS API below
component	elementType	'ul'	The component used for the root node. Either a string to use a DOM elemer
dense	bool	false	If `true`, compact vertical padding designed for keyboard and mouse inpu components as the `dense` context.
disablePadding	bool	false	If `true`, vertical padding will be removed from the list.
subheader	node		The content of the subheader, normally `ListSubheader`.

The `ref` is forwarded to the root element.

Any other props supplied will be provided to the root element (native element).

```
import React from 'react';
import List from '@material-ui/core/List';
import ListItem from '@material-ui/core/ListItem';
import ListItemText from '@material-ui/core/ListItemText';

function App() {
  return (
    <>
      <p>test</p>
    </>
  );
}

export default App;
```

Let's start by adding some todos to our page

We've added a static list with 2 todo items

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

```
import React, { useState, useRef } from 'react';
import TextField from '@material-ui/core/TextField';
import Button from '@material-ui/core/Button';
import List from '@material-ui/core/List';
import ListItem from '@material-ui/core/ListItem';
import ListItemText from '@material-ui/core/ListItemText';

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

  // add new todo to state and clear the input
  function handleAddClick() {
    const newTodo = textFieldRef.current.value;
    if(newTodo.length > 0) {
      setTodos([...todos, newTodo]);
      textFieldRef.current.value = '';
    }
```

Import useRef from react

Removing todos

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

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

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

  // add new todo to state and clear the input
  function handleAddClick() {
    const newTodo = textFieldRef.current.value;
    if(newTodo.length > 0) {
      setTodos([...todos, newTodo]);
```

Import IconButton and DeleteIcon

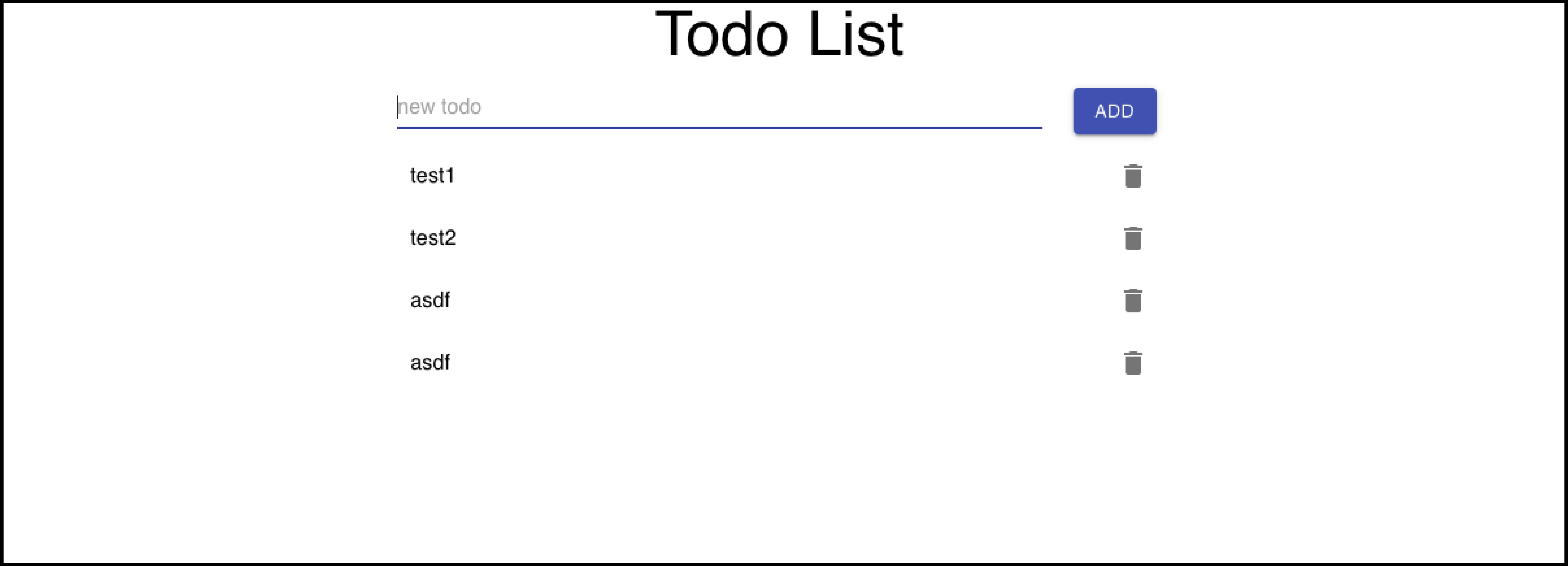
Responsive Layout

- 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 TextField from '@material-ui/core/TextField';
import Button from '@material-ui/core/Button';
import IconButton from '@material-ui/core/IconButton';
import DeleteIcon from '@material-ui/icons/Delete';
import Grid from '@material-ui/core/Grid';
import Typography from '@material-ui/core/Typography';
import List from '@material-ui/core/List';
import ListItem from '@material-ui/core/ListItem';
import ListItemText from '@material-ui/core/ListItemText';
import ListItemSecondaryAction from '@material-ui/core/ListItemSecondaryAction';

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

  // add new todo to state and clear the input
  function handleAddClick() {
    const newTodo = textFieldRef.current.value;
```

Import Grid and Typography

Desktop


Todo List

new todo

ADD

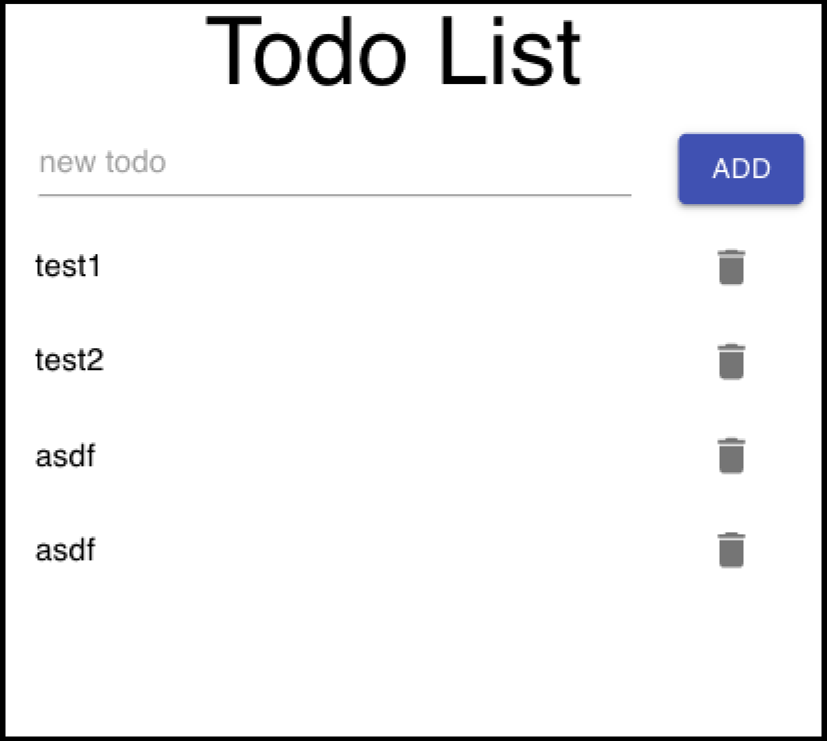
test1

test2

asdf

asdf

Phone


Todo List

new todo

test1

test2

asdf

asdf

Take your app further

- Refactoring
  - Make TodoItem its own component
- Ability to edit todos
- 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
- Write unit tests
  - Start here

Thank you!