TriValley Coders

Events Project

TriValley coders

Site Demo

Tech Stack

TriValley coders

Tech Stack

React	• Material UI • React Router • date-fns • Ramda	• Redux • React Redux • Redux Form • Redux Thunk
Express	• AWS SDK • MongoDB Node Driver • Passport • Helmet (to be added)	• Ramda
MongoDB	• Locally in development • MongoDB Atlas for production	

TriValley coders

Material Design

A “design language”

A design language or design vocabulary is an overarching scheme or style that guides the design of a complement of products or architectural settings.

- Wikipedia

Material UI

- Material UI is a set of React components that implement Google's Material Design
- Saves you a lot of work
- Excellent layout grid component
- Add consistency to your UI
- Is highly customizable
- New
  - Version 1.0 on 17-May-18
  - Currently v3.1.1 and evolving fast
- I enjoy their styling & customization model

React Router for Navigation

- React Router is a set of React components.
- Your components are displayed based on the path in the browser's address bar.
- There are web and native versions

React Router Code Example

date-fns

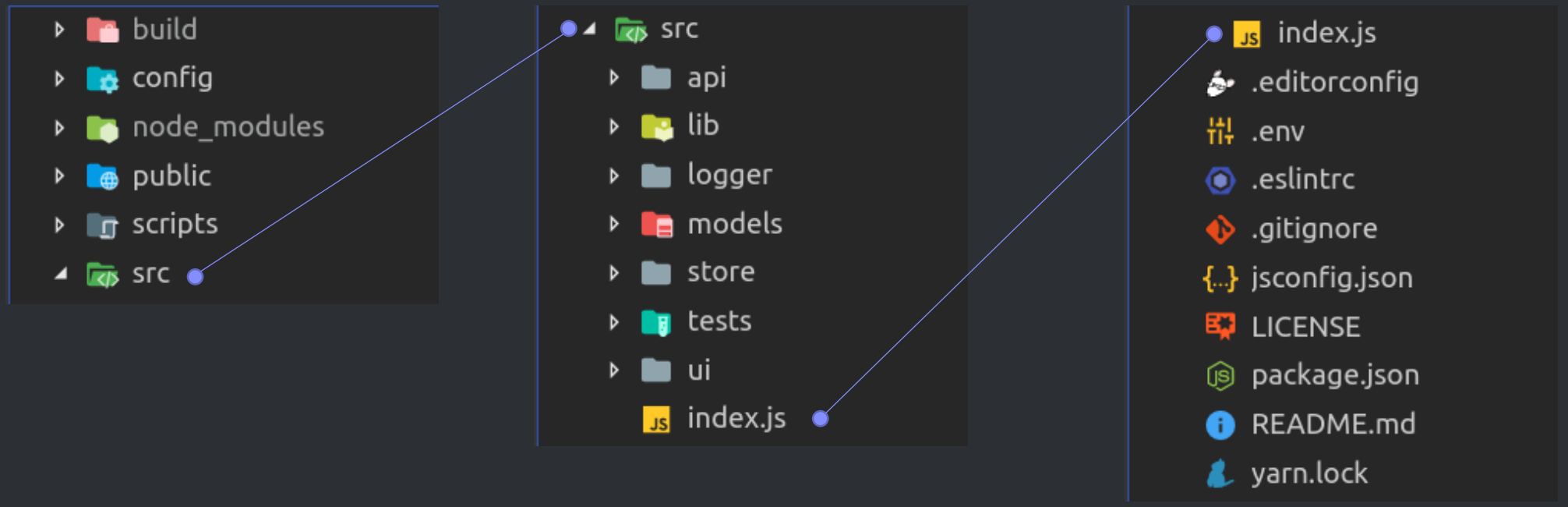
- JavaScript programmers need help with dates
- Date functions
- Light weight
  - Moment

Ramda

- Immutability library with a functional style
- Large library of functions for working with lists/arrays and objects
- Evetns only users 6 or 7 Ramda functions

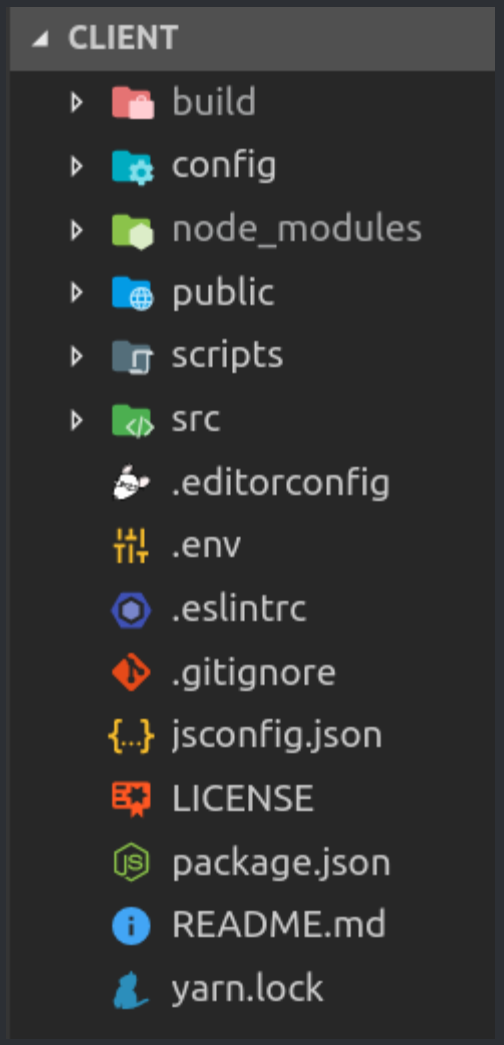
Project Structure

Top Level Folder Structure



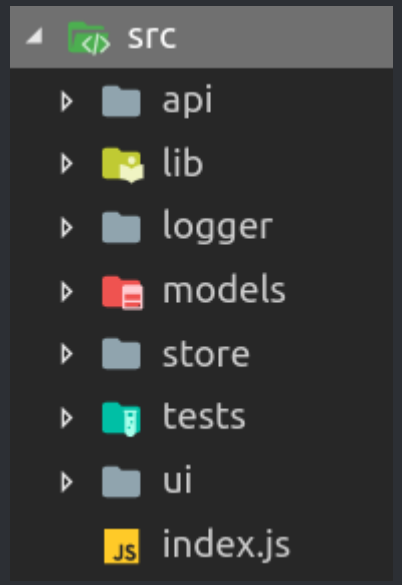
TriValley coders

Root Directory

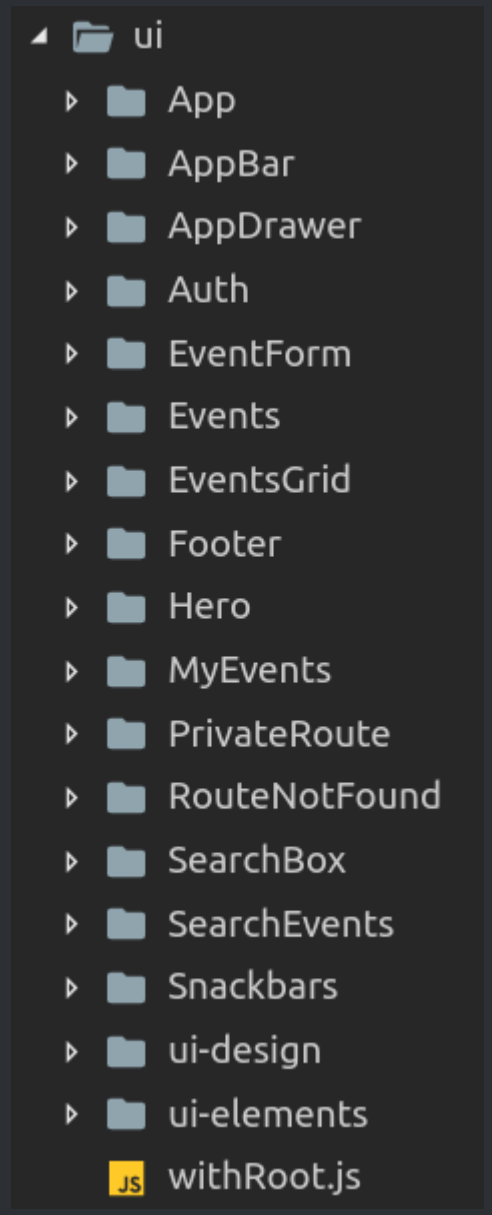


- build – production build
- config – for test, Webpack, etc.
- public – fairly empty html file
- scripts – Create React App scripts
- src – application code where most work is done
- .editorconfig – VS Code config settings
- .env – environment variables
- .eslintrc – custom ESLint rules
- .gitignore – rules for Git
- jsconfig.json – more VS Code config
- LICENSE – MIT open source
- package.json – npm configuration
- README
- yarn.lock – used by Yarn to manage package versions

/src Directory



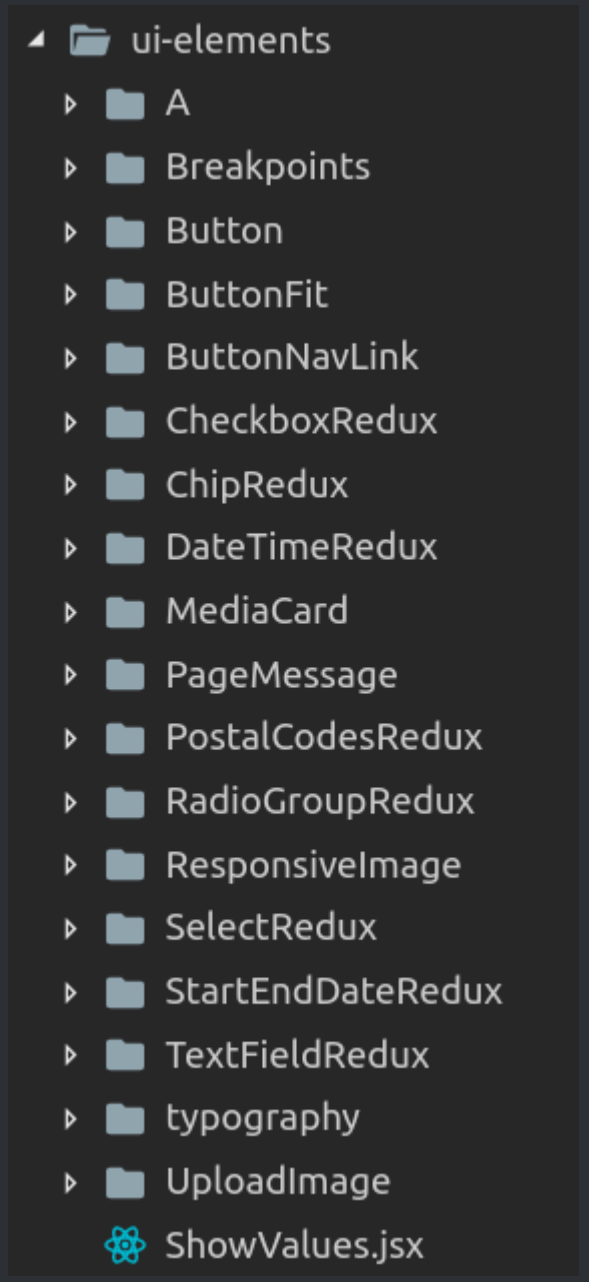
- api – all API calls are made from files in this directory
- lib – utility functions used throughout the project
- logger – an alternative to console.log() that produces colored output to the browser's console
- models – not in use
- store – the Redux store
- test – test suite
- ui – all UI components and where 95% of code is written
- index.js – main entry point for app



/ui Directory

- App – main app director, i.e., App.jsx
- AppBar – bar at top of page
- AppDrawer – menu slides out from left side
- Auth – components used for registration and authorization (register, login, etc.)
- EventForm – create, edit and view details of an event
- Events – container for EventsGrid
- EventsGrid – grid of events displayed on the home page
- Footer – the footer
- Hero – top section of home page that has search box
- MyEvent – view of events for the currently logged-in user
- PrivateRoute – higher order component that wraps Route and blocks access unless user is logged-in
- RouteNotFounc – used only in development
- SearchBox – component for search input
- SearchEvents – grid of events that are the result of a search
- Snackbars – temporary ui that shows results of user actions e.g., ‘Events loaded’
- ui-design – contains components that display colors and typography
- ui-elements – common components such as Button used through the app
- withRoot.js – contains the project theme

TriValley coders

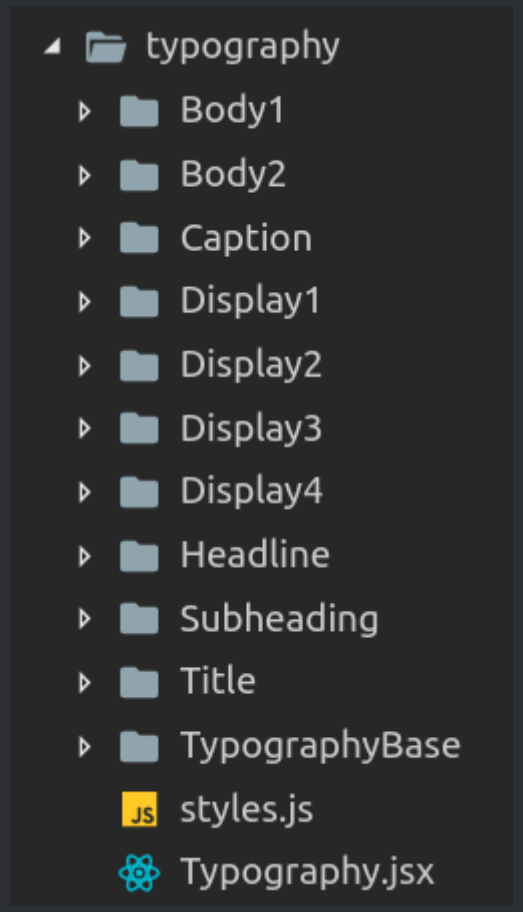


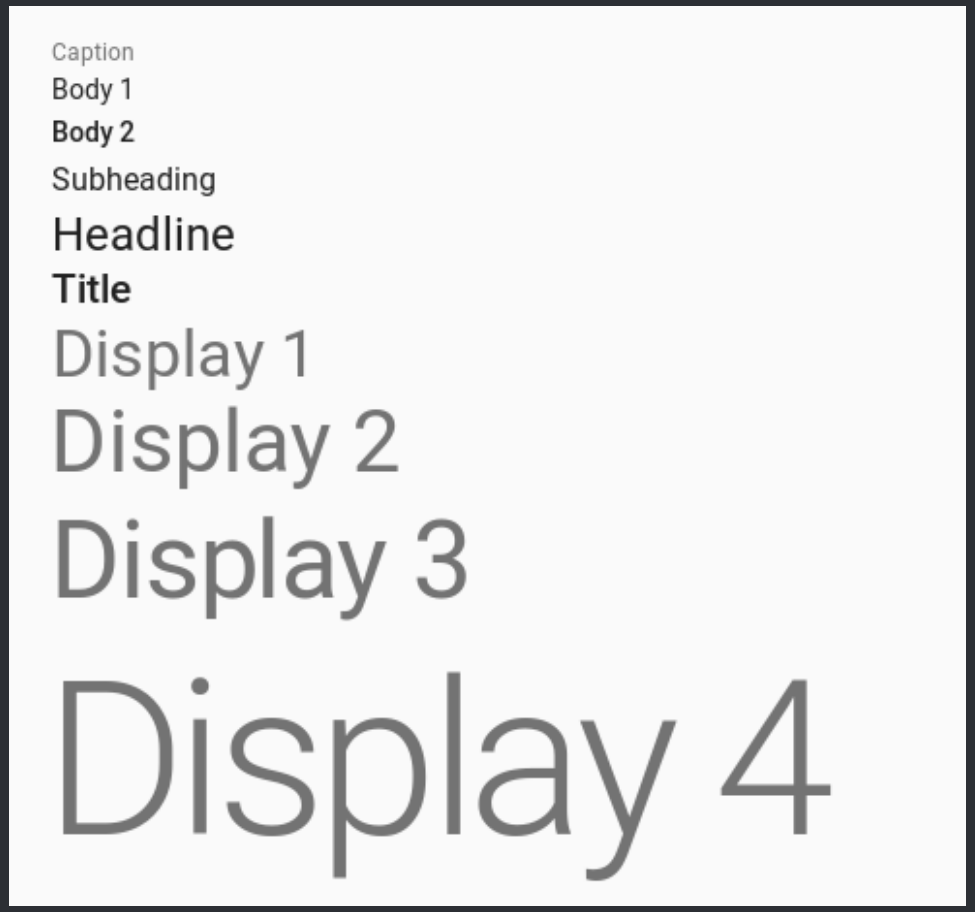
/ui-elements Directory

- Why?
  - Reuse of common components
  - Encapsulate styling and or desired functionality
  - Reduce redundant code

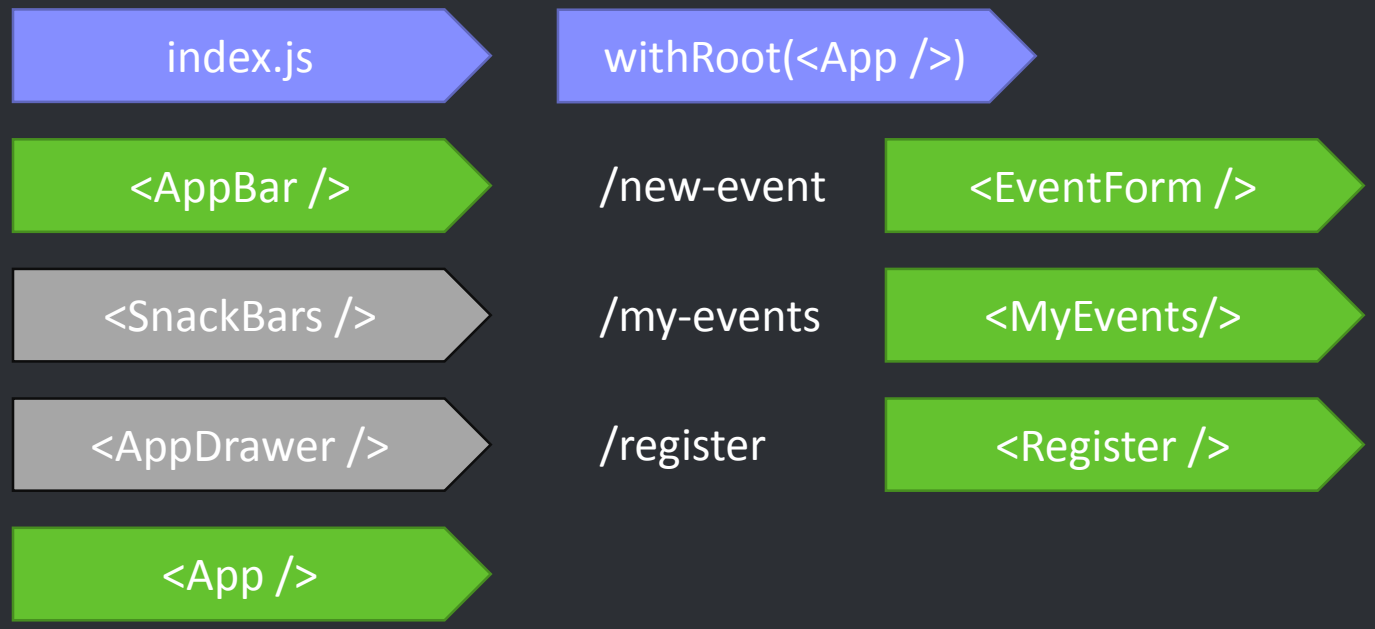
- A (<a>), Button – encapsulate styling
- ChipRedux, RadioGroupRedux, etc
  - Wrap Material UI components with Redux Form
- typography – components for displaying text
  - Explained later
- UploadImage – used on EventForm to upload images while creating/editing events

/src/ui-elements/typography





Project Startup



Routing

index.js

```
10  // import Footer from 'ui/Footer'
11
12
13  ReactDOM.render(
14    <Provider store={configureStore()}>
15      <Router>
16        <React.Fragment>
17          <AppBar />
18          <Snackbars />
19          <AppDrawer />
20          <Route component={App} />
21        </React.Fragment>
22      </Router>
23    </Provider>,
24    document.getElementById('root')        You, 4 m
25  )
26
```

App.jsx

```
71                ? <Hero />
72                : null
73            }
74            <div className={classes.body}>
75              <Switch>
76                <Route exact path='/typography' component={TypographyGuide} />
77                <Route exact path='/palette' component={Palette} />
78                <Route exact path='/new-event' component={EventForm} />
79                <PrivateRoute exact path='/new-event/:_id' component={EventForm} />
80                <Route exact path='/register' component={RegisterForm} />
81                <Route exact path='/login' component={LoginForm} />
82                <PrivateRoute exact path='/settings' component={SettingsForm} />
83                <Route exact path='/search-events/:searchValue' component={SearchEvents} />
84                <PrivateRoute exact path='/my-events' component={Events} />
85                <Route exact path='/events' component={Events} />
86                <Route exact path='/' component={Events} />
87                <Route component={RouteNotfound} />
88              </Switch>
89            </div>
90          </div>
91        </Fragment>
92      )
93    }
94  }
95
```

TriValley coders

Basic Componant Anatomy (functional component)

AnatomyFunction.jsx

```
1   import React from 'react'
2   import { withStyles } from '@material-ui/core/styles'
3
4   const Anatomy = ({ classes }) => {
5
6     return (
7       <div className={classes.wrapper}>
8         ...
9       </div>
10    )
11  }
12
13  const styles = theme => ({
14    wrapper: {
15      backgroundColor: 'blue'
16    }
17  })
18
19  export default withStyles(styles)(Anatomy)
```

1. Import withStyles()
2. Create a ‘styles’ object optionally passing it the theme
3. Write the style as an object using camelCase properties
4. ‘classes’, an object, is passed in as a prop
5. Use classes.propertyName inside of className={}

TriValley coders

Basic Componant Anatomy (class component)

AnatomyClass.jsx

```jsx
import React from 'react'
import { withStyles } from '@material-ui/core/styles'

class Anatomy extends React.Component {

  render() {

    const { classes } = this.props
    return (
      <div className={classes.wrapper}>
        ...
      </div>
    )
  }

}

const styles = theme => ({
  wrapper: {
    backgroundColor: 'blue'
  }
})

export default withStyles(styles)(Anatomy)
```

TriValley coders

Debugging