

### Setup

Create your react application project folders and files:

```
npx create -re act-app <pr oje ct_ nam e>
```

Navigate into your newly created project folder

```
cd <pr oje ct_ nam e>
```

Launch VS Code (or preferred IDE) using `code .` and press Enter

In your terminal window, install the Material UI package:

```
npm install @mater ial -ui /core
```

### Add Fonts & Icons

Material UI defines Roboto as the default font. Edit `index.html` and add the following statement inside your `<head>` block:

```
<link rel="st yle she et" href="h ttp s:/ /fo nts.go ogl eap is.c om /cs s?f ami ly= Rob oto :30 0,4 00, 500 ,70 0& dis pla y=s wap " />
```

Modify the global style sheet `index.css` and place Roboto in the first position (see below)

```
body { ... font-f amily: 'Roboto', -apple -sy stem, BlinkM acs yst emFont, 'Segoe UI', 'Oxygen', ... }
```

Add a link for the FONT Icons. Place it just below the 'Roboto' link

```
<link rel="st yle she et" href="h ttp s:/ /fo nts.go ogl eap is.c om /ic on? fam ily =Ma ter ial +Ic ons " />
```

Install SVG icons:

```
npm install @mater ial -ui /icons
```

### Default Styles and use of the App component.

Since Material-UI provides a default theme of baseline components, styles, colors, and typography, we will not need the definitions found in the `App.css` created during the initial setup.

**Edit the `App.css` file and clear its contents.**

By convention for React development, your App component should be placed under the `.src` directory.

- 1) Create a folder named 'App' in your `src` director
- 2) Next move your 'App.\*' files into this directory.
- 3) Modify the import statement in your `index.js` file to correctly find your `App.js` file.  
(ex. `import App from './App /App';`)

### Baseline Component Wrappers

Material UI promotes the functional component design. Although it also support classed based component design. The implementation is 'different' and can be confusing if you are just getting started.

An easy way to handle this is to create wrappers for all of the baseline components that implement the function component design. This way you can use your wrapper based components consistently no matter which component design method you want to use.

- 1) Create a subfolder named "components" in your `src` directory.
- 2) Inside this new folder, create another folder named 'controls'.
- 3) Inside your controls folder, you will create a JS/JSX file for every baseline component that you wish to use (ex: 'button', 'input', 'select', 'datepicker', etc.).
- 4) Also, to make importing these items easier and to keep your code 'clean', create a `controls.js` file.



### Sample Button wrapper

```
import React from 'react'

import { Button as MuiButton, makeStyles } from '@material-ui/core'
const useStyles = makeStyles(( theme ) => ({
  root: {
    margin: theme.spacing(0.5)
  },
  label: {
    textTransform: "none"
  }
}))

const Input= (props) => {
  const { text, size, color, variant, onClick, ...other} = props
  const classes = useStyles();
  return (
    <MuiButton
      variant={ variant || "contained" }
      size={size || "small"}
      color={color || "primary"}
      onClick={onClick}
      {...other}
      classes={{ root:classes.root, label: classes.label }}>
      {text}
    </MuiButton>
  )
}

export default Input
```

### Example use of this component

```
<Controls.Button
  text="Cancel"
  onClick={ handleCancel }
/>
```

This example demonstrates how to "wrap" around the baseline button component. It provides "default" variant, size and color properties. The only props that you need to supply will be the text and the click handler.

Note: the `...other` parameter will allow to make use of ALL available properties.

Note: In order to prevent naming colisions, the 'Button' import from Material UI was given an alias of 'MuiButton'. Thus we can now call our wrapped component 'Button'.



By **Cash** (CashM)  
[cheatography.com/cashm/](https://cheatography.com/cashm/)

Published 16th May, 2021.  
Last updated 16th May, 2021.  
Page 2 of 3.

Sponsored by **Readable.com**  
Measure your website readability!  
<https://readable.com>

### Control.JS File

```
import Button from './Button'  
import Input from './Input'  
import Select from './Select'  
const Controls = {  
  Button,  
  Input,  
  Select  
}  
export default Controls;  
---  
Now in your other components, simply import everything using the following:  
import Controls from '.../.../components/controls/Controls';
```

This file will make importing your wrapped baseline components easier and will help to keep your code 'clean'.

For each component in your controls folder, add an import statement and the component name in the const definition.



By **Cash** (CashM)  
[cheatography.com/cashm/](https://cheatography.com/cashm/)

Published 16th May, 2021.  
Last updated 16th May, 2021.  
Page 3 of 3.

Sponsored by **Readable.com**  
Measure your website readability!  
<https://readable.com>