Using Breakpoints and Media Queries in Material-UI

How to make your app responsive using Material-UI

Chad Murobayashi Follow

Feb 28, 2021 · 5 min read ★

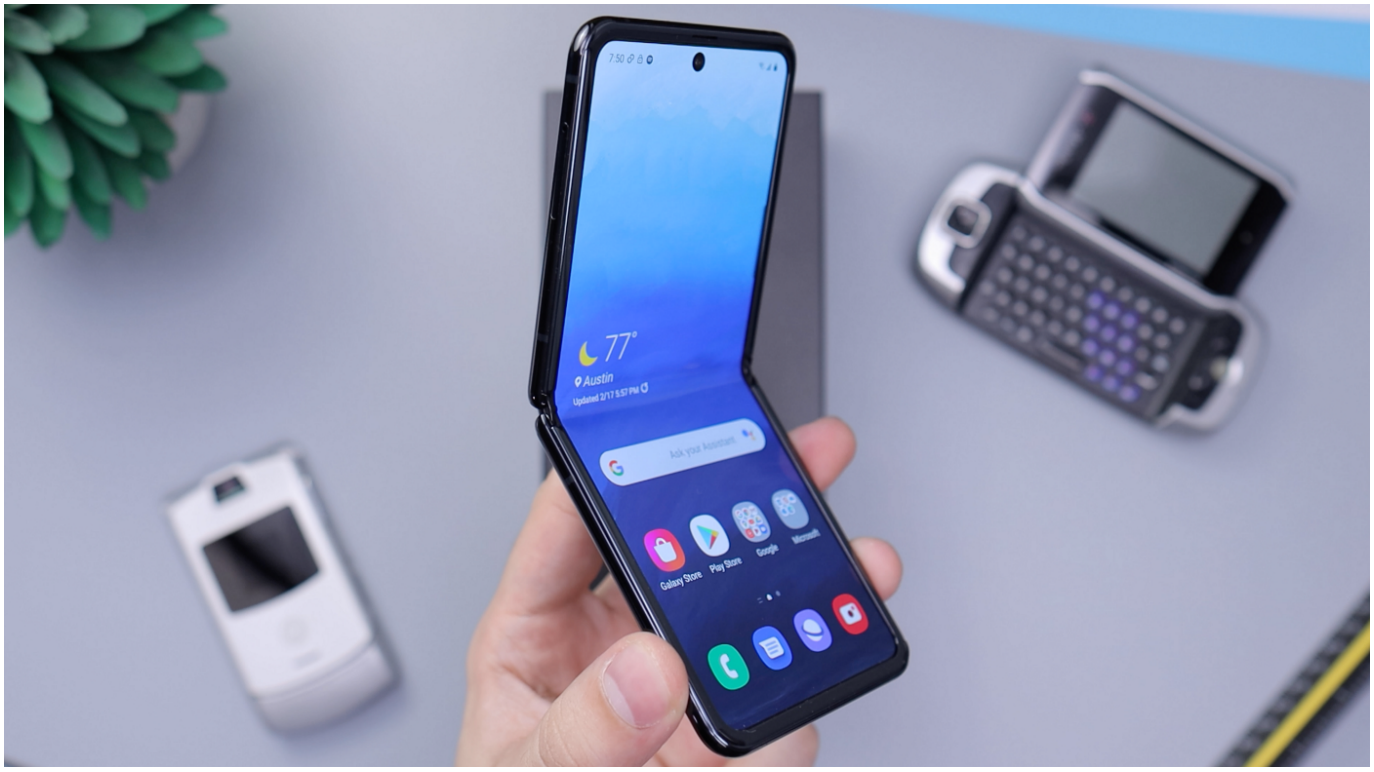


Photo by Daniel Romero on Unsplash

Material-UI is one of the most popular React UI component libraries. It uses Google's Material Design.

I have been having a lot of fun using Material-UI for my recent projects. It is now my go-to for styling a React application. It took me a while to understand the best way to write media queries, so I wrote this article to help others in the future.

There are a few different ways we can create a responsive app using Material-UI. In this article, we will take a look at using breakpoints to write media queries using `makeStyles` and the `useMediaQuery` hook.

. . .

Understanding Breakpoints

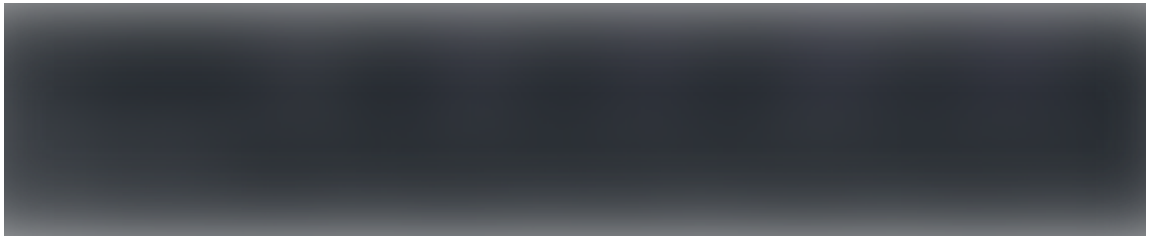
Before we dive into writing media queries in Material-UI, let's get an understanding of the use of breakpoints.

> *"For optimal user experience, material design interfaces need to be able to adapt their layout at various breakpoints. Material-UI uses a simplified implementation of the original specification."*

Material-UI comes with a default theme, including breakpoints. The default breakpoints are as follows.

- **xs,** extra-small: 0px
- **sm,** small: 600px

- **md**, medium: 960px
- **lg**, large: 1280px
- **xl**, extra-large: 1920px



The theme provider provides four styles helpers for us to write media queries. You can read more about how to implement each one in the links below.

- theme.breakpoints.up(key)
- theme.breakpoints.down(key)
- theme.breakpoints.only(key)
- theme.breakpoints.between(start, end)

Basically, we will pass in a breakpoint as an argument, and this will return a media query for us.

For example, if we can use `theme.breakpoints.up('sm')`. This means that we want to match screen widths greater than or equal to the small screen size (600px).

You can also customize the breakpoints to suit your needs. To do so, you will need to overwrite the default values with the `createMuiTheme` function.

Breakpoints with makeStyles

Now that we have an understanding of breakpoints, let's put them to use. The first way to use them is by writing media queries using `makeStyles`. You can check out the documentation here.

First, we need to import `makeStyles` from Material-UI.

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

The `makeStyles` hook takes either a function or an object. In order for us to use the theme and breakpoints, we will provide a function with the theme as a prop. In this function, we will pass an object with classes. Each class has styles that we want to apply to our project.

The `makeStyles` function returns a hook. This hook is often referred to as `useStyles`.

In the example below, we are creating a root class with a height of 100vh and a background color of blue. We can use the default theme breakpoints to write media queries.

We are saying that, whenever the screen size is small size (600px) or above, the background color will be red. When it is medium size (960px) or above, the background color will be green. When it is large size (1280px) or above, the background color will be orange. And when it is extra-large size (1920px) or above, the background color will be cyan.

```
const useStyles = makeStyles(theme => ({
  root: {
    height: '100vh',
```

```
    backgroundColor: 'blue',
    [theme.breakpoints.up('sm')]: {
      backgroundColor: 'red',
    },
    [theme.breakpoints.up('md')]: {
      backgroundColor: 'green',
    },
    [theme.breakpoints.up('lg')]: {
      backgroundColor: 'orange',
    },
    [theme.breakpoints.up('xl')]: {
      backgroundColor: 'cyan',
    },
  },
}));
```

To use the `useStyles` hook, we will call it in our application and `makeStyles` will generate all of the styles for us and store it in an object. Generally, this object will be called `classes`.

Then we can add `classNames` to our elements and add the class name of `classes.root` to apply the styles and media queries.

```
const App = () => {
  const classes = useStyles();

  return (
```

```
      <div className={classes.root}>
        <Typography>
          The background will change color based on the screen size.
        </Typography>
      </div>
    );
};
```

Now, the background color will change according to the screen size.

Using the useMediaQuery Hook

In some cases, adding media queries to the styling is not enough. For example, this can get complicated if you want to render something on a page conditionally. In this case, you can use the `useMediaQuery` hook.

According to the documentation,

> *"This is a CSS media query hook for React. It listens for matches to a CSS media query. It allows the rendering of components based on whether the query matches or not."*

To use the `useMediaQuery` hook, first import it from Material-UI.

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

In the component, call the `useMediaQuery` hook and pass in a media query as the argument. This will return a true or false value.

```
const showText = useMediaQuery('(min-width:600px)');
```

We can also use the breakpoints as in the previous example. To do so, we will also need to import the `useTheme` hook and call it in the component.

```
import { useTheme, useMediaQuery } from '@material-ui/core';
```

When we call the `useTheme` hook, it will return an object with all of the theme properties, including breakpoints. Now, we can use the styles helpers like before.

```
const theme = useTheme();
const showText = useMediaQuery(theme.breakpoints.up('sm'));
```

Next, we can use the `showText` variable to conditionally render an element based on the screen size. If the screen is smaller than 600px (the default small size), then `showText` will be false. If it is 600px or larger, it will be true.

```
const App = () => {
  const theme = useTheme();
  const showText = useMediaQuery(theme.breakpoints.up('sm'));


  return (
    <div>
      {showText && <Typography variant="h1">Appear when 600px and
above</Typography>}
    </div>
  );
};
```

Now, when the screen size goes below 600px, `showText` will be false and the test will not render. When the screen size is 600px and above, it will show.

You can see the full code for using both the `makeStyles` and `useMediaQuery` in the example below.

. . .

Conclusion

Thanks for reading! I hope this article was helpful for you to understand how to use breakpoints and media queries with Material-UI.

You can easily add media queries using `makeStyles`. If you need to conditionally render something on the page, use the `useMediaQuery` hook.

If you'd like to learn more about Material-UI, check out the two articles below.

Create a Customized Color Theme in Material-UI

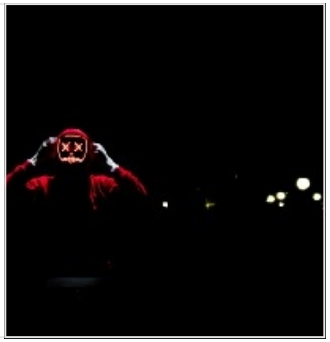
Making a custom color theme for your next React project is easy

medium.com

How to Apply Dark Mode Using Material-UI

Turn your React applications from light mode to dark mode with just one switch

medium.com



JavaScript React Material Ui Media Queries Programming

WRITTEN BY

Chad Murobayashi

Follow

Software Engineer living in Tokyo, Japan. Born and raised in Hawaii.

Level Up Coding

Follow

Coding tutorials and news. The developer homepage gitconnected.com && skilled.dev

More From Medium

React: Best Practices

Akshata Waghe in Udgama Blog

Writing Clean Code in JavaScript

Bhagya Vithana in Bits and Pieces

Chart.js With Bootstrap For Absolute Beginners

Nisal Suranaka

Highway, the only modern & flexible JS transitions manager you'll ever need

Anthony Du Pont in Dogstudio

Authorization in React with MOBX

Aleksandar Rajic

Porting From Nashorn: How to Handle JS Multi-Threading on GraalVM

Andreas Müller in The Startup

State management with React Context, TypeScript, and GraphQL

Lily Barrett in HackerNoon.com

What's New in Material-UI Version 5

Chad Murobayashi in JavaScript in Plain English

Learn more.

Medium is an open platform where 170 million readers come to find insightful and dynamic thinking. Here, expert and undiscovered voices alike dive into the heart of any topic and bring new ideas to the surface. Learn more

Make Medium yours.

Follow the writers, publications, and topics that matter to you, and you'll see them on your homepage and in your inbox. Explore

Write a story on Medium.

If you have a story to tell, knowledge to share, or a perspective to offer — welcome home. It's easy and free to post your thinking on any topic. Start a blog

Medium

About Write Help Legal