

# Typography

The theme provides a set of type sizes that work well together, and also with the layout grid.

## Font family


You can change the font family with the `theme.typography.fontFamily` property.

For instance, this example uses the system font instead of the default Roboto font:

```
const theme = createTheme({
  typography: {
    fontFamily: [
      '-apple-system',
      'BlinkMacSystemFont',
      '"Segoe UI"',
      'Roboto',
      '"Helvetica Neue"',
      'Arial',
      'sans-serif',
      '"Apple Color Emoji"',
      '"Segoe UI Emoji"',
      '"Segoe UI Symbol"',
    ].join(','),
  },
});
```

## Self-hosted fonts

To self-host fonts, download the font files in `ttf`, `woff`, and/or `woff2` formats and import them into your code.

 This requires that you have a plugin or loader in your build process that can handle loading `ttf`, `woff`, and `woff2` files. Fonts will *not* be embedded within your bundle. They will be loaded from your webserver instead of a CDN.

```
import RalewayWoff2 from './fonts/Raleway-Regular.woff2';
```

Next, you need to change the theme to use this new font. In order to globally define Raleway as a font face, the [CssBaseline](#) component can be used (or any other CSS solution of your choice).

```
import RalewayWoff2 from './fonts/Raleway-Regular.woff2';

const theme = createTheme({
  typography: {
    fontFamily: 'Raleway, Arial',
  },
  components: {
    MuiCssBaseline: {
      styleOverrides: `
```

```

    @font-face {
      font-family: 'Raleway';
      font-style: normal;
      font-display: swap;
      font-weight: 400;
      src: local('Raleway'), local('Raleway-Regular'), url(${RalewayWoff2})
format('woff2');
      unicodeRange: U+0000-00FF, U+0131, U+0152-0153, U+02BB-02BC, U+02C6,
U+02DA, U+02DC, U+2000-206F, U+2074, U+20AC, U+2122, U+2191, U+2193, U+2212, U+2215,
U+FEFF;
    }
  },
},
});

// ...
return (
  <ThemeProvider theme={theme}>
    <CssBaseline />
    <Box
      sx={{
        fontFamily: 'Raleway',
      }}
    >
      Raleway
    </Box>
  </ThemeProvider>
);

```

Note that if you want to add additional `@font-face` declarations, you need to use the string CSS template syntax for adding style overrides, so that the previously defined `@font-face` declarations won't be replaced.

## Font size

MUI uses `rem` units for the font size. The browser `<html>` element default font size is `16px`, but browsers have an option to change this value, so `rem` units allow us to accommodate the user's settings, resulting in a better accessibility support. Users change font size settings for all kinds of reasons, from poor eyesight to choosing optimum settings for devices that can be vastly different in size and viewing distance.

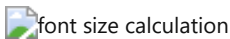
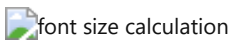
To change the font-size of MUI you can provide a `fontSize` property. The default value is `14px`.

```

const theme = createTheme({
  typography: {
    // In Chinese and Japanese the characters are usually larger,
    // so a smaller fontsize may be appropriate.
    fontSize: 12,
  },
});

```

The computed font size by the browser follows this mathematical equation:



## Responsive font sizes

The `theme.typography.* variant` properties map directly to the generated CSS. You can use [media queries](#) inside them:

```
const theme = createTheme();

theme.typography.h3 = {
  fontSize: '1.2rem',
  '@media (min-width:600px)': {
    fontSize: '1.5rem',
  },
  [theme.breakpoints.up('md')]: {
    fontSize: '2.4rem',
  },
};
```

{{"demo": "CustomResponsiveFontSizes.js"}}

To automate this setup, you can use the [responsiveFontSizes\(\)](#) helper to make Typography font sizes in the theme responsive.

{{"demo": "ResponsiveFontSizesChart.js", "hideToolbar": true}}

You can see this in action in the example below. Adjust your browser's window size, and notice how the font size changes as the width crosses the different [breakpoints](#):

```
import { createTheme, responsiveFontSizes } from '@mui/material/styles';

let theme = createTheme();
theme = responsiveFontSizes(theme);
```


{{"demo": "ResponsiveFontSizes.js"}}

## Fluid font sizes

To be done: [#15251](#).

## HTML font size

You might want to change the `<html>` element default font size. For instance, when using the [10px simplification](#).

 *Changing the font size can harm accessibility [🔗](#). Most browsers agreed on the default size of 16px, but the user can change it. For instance, someone with an impaired vision could have set their browser's default font size to something larger.*

The `theme.typography.htmlFontSize` property is provided for this use case, which tells MUI what the font-size on the `<html>` element is. This is used to adjust the `rem` value so the calculated font-size always match the

specification.

```
const theme = createTheme({
  typography: {
    // Tell MUI what's the font-size on the html element is.
    htmlFontSize: 10,
  },
});
```

```
html {
  font-size: 62.5%; /* 62.5% of 16px = 10px */
}
```

*You need to apply the above CSS on the html element of this page to see the below demo rendered correctly*

{{"demo": "FontSizeTheme.js"}}

## Variants

The typography object comes with [13 variants](#) by default:

- h1
- h2
- h3
- h4
- h5
- h6
- subtitle1
- subtitle2
- body1
- body2
- button
- caption
- overline

Each of these variants can be customized individually:

```
const theme = createTheme({
  typography: {
    subtitle1: {
      fontSize: 12,
    },
    body1: {
      fontWeight: 500,
    },
    button: {
      fontStyle: 'italic',
    },
  },
});
```

```
{{"demo": "TypographyVariants.js"}}
```

## Adding & disabling variants

In addition to using the default typography variants, you can add custom ones, or disable any you don't need. Here is what you need to do:

### Step 1. Update the theme's typography object

```
const theme = createTheme({
  typography: {
    poster: {
      color: 'red',
    },
    // Disable h3 variant
    h3: undefined,
  },
});
```

### Step 2. Update the necessary typings (if you are using TypeScript)

*If you aren't using TypeScript you should skip this step.*

You need to make sure that the typings for the theme's `typography` variants and the `Typography`'s `variant` prop reflects the new set of variants.

```
declare module '@mui/material/styles' {
  interface TypographyVariants {
    poster: React.CSSProperties;
  }

  // allow configuration using `createTheme`
  interface TypographyVariantsOptions {
    poster?: React.CSSProperties;
  }
}

// Update the Typography's variant prop options
declare module '@mui/material/Typography' {
  interface TypographyPropsVariantOverrides {
    poster: true;
    h3: false;
  }
}
```

### Step 3. You can now use the new variant

```
{{"demo": "TypographyCustomVariant.js", "hideToolbar": true}}
```

```
<Typography variant="poster">poster</Typography>;
```

```
/* This variant is no longer supported */  
<Typography variant="h3">h3</Typography>;
```

## Default values

You can explore the default values of the typography using [the theme explorer](#) or by opening the dev tools console on this page ( `window.theme.typography` ).