

文字铸排

主题会提供一套能够一起协调工作的类型大小，也提供了布局网格。

Font family

您可以使用 `theme.typography.fontFamily` 属性来更改 font family。

例如，这个例子使用是系统字体而不是默认的 Roboto 字体：

```
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(','),
  },
});
```

自托管的字体

若想使用自托管的字体，请下载 `ttf`，`woff`，以及/或者 `woff2` 格式的字体文件，然后将它们导入到你的代码中去。

⚠ 这则需要在你的生成过程中有一个插件或者加载器，用它们可以处理 `ttf`，`woff` 和 `woff2` 文件的加载。字体将不会内嵌入你的资源文件包（bundle）。它们将从您的网络服务器上而不是 CDN 中加载。

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

接下来，您需要做的是修改主题，来使用这一个新的字体。如果想在全局定义 Raleway 作为一个字体，您可以使用 [CssBaseline](#) 组件（或者你也可以选择你想要的任意其他 CSS 方案）。

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

字体大小 (Font size)

Material-UI 使用 `rem` 单元来定义字体的大小。浏览器 `<html>` 元素的默认字体大小为 `16px`，但是浏览器提供了一个改变这个值的选项，所以 `rem` 单元能够让我们适应用户的设置，从而提供更好的无障碍设计的支持。其实用户改变字体大小设置的原因多种多样，有不太好的视力，或者选择适应设备的最佳设置，这样在大小和查看距离上会有很大的差异。

若想更改 Material-UI 的字体大小，您可以提供一个 `fontSize` 属性。它的默认值为 `14px`。

```
const theme = createTheme({
  typography: {
    // 中文字符和日文字符通常比较大，
    // 所以选用一个略小的 fontSize 会比较合适。
    fontSize: 12,
  },
});
```

浏览器计算出来的字体大小遵循了以下数学方程式：



计算字体大小



计算字体大小

响应的字体大小

`theme.typography.*` [variant](#) 属性会直接映射到生成的 CSS。您可以在当中使用 [媒体查询 \(media queries\)](#)：

```
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"}}
```

若你想实现此设置的自动化，则可以使用 [responsiveFontSizes\(\)](#) 的帮助程序将 Typography 的字体大小在主题设置为响应性。

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

您可以在下面的示例中看到这个操作。请尝试调整浏览器的窗口大小，您可以注意到当切换到不同的 [breakpoints](#) 设置的宽度，字体的大小也随之改变。

```
import { createTheme, responsiveFontSizes } from '@material-ui/core/styles';

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



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

流式文字大小

待完成: [#15251](#)。

HTML 的字体大小

您可能想要更改 `<html>` 元素的默认字体大小。例如，当您使用 [10px 简化](#) 时。

 *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 Material-UI what's the font-size on the html element is.
    htmlFontSize: 10,
  },
});

htmlFontSize: 10,
},
});

htmlFontSize: 10,
},
});
```

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

您需要在此页面的 `html` 元素上应用上述的 CSS 才能看到以下演示正确的渲染了。

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

变体

默认情况下，typography object 为带有 [13 种变体](#)：

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

每个变体都可以被单独地定制:

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

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

添加 & 禁用变体

除了使用默认的排版变体外，你还可以添加自定义的排版，或者禁用任何你不需要的排版。 Here is what you need to do:

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

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

Step 2. Step 2. Step 2. Step 2. 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 '@material-ui/core/styles' {
  interface TypographyVariants {
    poster: React.CSSProperties;
  }

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

// Update the Typography's variant prop options
declare module '@material-ui/core/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>;
```

默认值

您可以使用 [主题探索功能](#)，或者在此页面上打开 dev 工具控制（`window.theme.typogry`）来查看 `typography` 的默认值。