Components 组件

The theme's `components` key allows you to customize a component without wrapping it in another component. You can change the styles, the default props, and more. 你可以更改样式、默认属性等等其他东西。 你可以更改样式、默认属性等等其他东西。

重写全局样式

You can change the default of every prop of a MUI component. A `defaultProps` key is exposed in the theme's `components` key for this use case.

```
const theme = createTheme({
  components: {
    // Name of the component
    MuiButtonBase: {
      defaultProps: {
        // The props to change the default for.
        disableRipple: true, // No more ripple!
      },
    },
  },
});
```

{{"demo": "DefaultProps.js"}}

在组件 API 页面的 **CSS**介绍部分，列出了每个组件样式类的文档。

默认属性

要用 TypeScript 重写 lab 组件的样式，请查看 此文档。

```
const theme = createTheme({
  components: {
    // Name of the component
    MuiButton: {
      styleOverrides: {
        // Name of the slot
        root: {
          // Some CSS
          fontSize: '1rem',
        },
      },
    },
  },
});
```

{{"demo": "GlobalThemeOverride.js"}}

The list of each component's classes is documented under the **CSS** section of its API page.

要使用 TypeScript 覆盖实验室组件样式，请检查 此文档

Overrides based on props

You can change the default of every prop of a MUI component. 下面示例就是指定 `defaultProps` 属性覆盖 `components` 下组件的默认属性。

The `ownerState` prop is a combination of public props that you pass to the component + internal state of the component.

```
const theme = createTheme({
  components: {
    MuiButton: {
      variants: [
        {
          props: { variant: 'dashed' },
          style: {
            textTransform: 'none',
            border: `2px dashed grey${blue[500]}`,
          },
        },
        {
          props: { variant: 'dashed', color: 'secondary' },
          style: {
            border: `4px dashed ${red[500]}`,
          },
        },
      ],
    },
  },
});
```

如果你用的是 TypeScript，那么需要使用moduleaugmentation定义新的 variants/colors

Using `sx` (experimental) syntax

If you are not familiar `sx`, first check out the concept and the difference with the `styled`.

覆盖所有组件实例的另一种方式是调整 theme configuration variables。

{{"demo": "GlobalThemeOverrideSx.js"}}

添加新的组件变量

You can use the `variants` key in the theme's `components` section to add new variants to Material-UI components. These new variants can specify what styles the component should have when specific props are applied. 这些变量能够决定当使用特定的属性时组件应该采用什么样的样式。 这些变量能够决定当使用特定的属性时组件应该采用什么样的样式。

在组件名称（如：MuiButton）下以数组形式定义组件变量。 数组中的每个变量都会对应一个 CSS 类添加到 HTML `<head>` 中。 For each of them a CSS class is added to the HTML `<head>`. The order is important, so make sure that the styles that should win are specified last.

```
declare module '@material-ui/core/Button/Button' {
  interface ButtonPropsVariantOverrides {
    dashed: true;
  }
}
```

If you're using TypeScript, you'll need to specify your new variants/colors, using module augmentation.

```
const theme = createTheme({
  components: {
    // Name of the component
    MuiButtonBase: {
      defaultProps: {
        // The props to change the default for.
        disableRipple: true, // No more ripple!
      },
    },
  },
});
        disableRipple: true, // No more ripple!
      },
    },
  },
});
```

{{"demo": "GlobalThemeVariants.js"}}

主题变量

Another way to override the look of all component instances is to adjust the theme configuration variables.

```
const theme = createTheme({
  typography: {
    button: {
      fontSize: '1rem',
    },
  },
});
```

{{"demo": "ThemeVariables.js"}}