Dark mode

MUI comes with two palette modes: light (the default) and dark.

Você pode deixar o tema escuro definindo `mode: 'dark'`.

```
const darkTheme = createTheme({
  palette: {
    mode: 'dark',
  },
});
```

While it's only a single value change, the `createTheme` helper modifies several palette values. The colors modified by the palette mode are the following:

{{"demo": "DarkTheme.js", "bg": "inline", "hideToolbar": true}}

> *Note: The colors are modified only if you use the default palette. If you have a custom palette, you need to make sure that you have the correct values based on the* `mode`*. The following section explains how you can do it.*

Dark mode with custom palette

The easiest way of how you can implement your custom palette that depends on mode is to have a dedicated function that will return the palette based on the mode. For example:

```
const getDesignTokens = (mode: PaletteMode) => ({
  palette: {
    mode,
    ...(mode === 'light'
      ? {
          // palette values for light mode
          primary: amber,
          divider: amber[200],
          text: {
            primary: grey[900],
            secondary: grey[800],
          },
        }
      : {
          // palette values for dark mode
          primary: deepOrange,
          divider: deepOrange[700],
          background: {
            default: deepOrange[900],
            paper: deepOrange[900],
          },
          text: {
            primary: '#fff',
            secondary: grey[500],
          },
        }),
  },
});
```

You can see on the example that there are different colors used based on whether the mode is light or dark. The next step is to use this function when creating the theme.

```
export default function App() {
  const [mode, setMode] = React.useState<PaletteMode>('light');
  const colorMode = React.useMemo(
    () => ({
      // The dark mode switch would invoke this method
      toggleColorMode: () => {
        setMode((prevMode: PaletteMode) =>
          prevMode === 'light' ? 'dark' : 'light',
        );
      },
    }),
    [],
  );

  // Update the theme only if the mode changes
  const theme = React.useMemo(() => createTheme(getDesignTokens(mode)), [mode]);

  return (
    <ColorModeContext.Provider value={colorMode}>
      <ThemeProvider theme={theme}>
        <Page />
      </ThemeProvider>
    </ColorModeContext.Provider>
  );
}
```

Here is a fully working example:

{{"demo": "DarkThemeWithCustomPalette.js", "defaultCodeOpen": false}}

Toggling color mode

You can use the React context to toggle the mode with a button inside your page.

{{"demo": "ToggleColorMode.js", "defaultCodeOpen": false}}

System preference

Usuários podem especificar uma preferência por um tema claro ou escuro. O método pelo qual o usuário expressa a sua preferência pode variar. Pode ser uma configuração de sistema exposta pelo Sistema Operacional, ou uma configuração controlada pelo Agente de Usuário.

Você pode utilizar essa preferência dinamicamente com o hook useMediaQuery e a consulta de mídia prefers-color-scheme.

Por exemplo, você pode ativar o modo escuro automaticamente:

```
import * as React from 'react';
import useMediaQuery from '@mui/material/useMediaQuery';
import { createTheme, ThemeProvider } from '@mui/material/styles';
import CssBaseline from '@mui/material/CssBaseline';

function App() {
  const prefersDarkMode = useMediaQuery('(prefers-color-scheme: dark)');

  const theme = React.useMemo(
    () =>
      createTheme({
        palette: {
          mode: prefersDarkMode ? 'dark' : 'light',
        },
      }),
    [prefersDarkMode],
  );

  return (
    <ThemeProvider theme={theme}>
      <CssBaseline />
      <Routes />
    </ThemeProvider>
  );
}
```