nativeTheme

Read and respond to changes in Chromium's native color theme.

Process: Main

Events

The `nativeTheme` module emits the following events:

Event: 'updated'

Emitted when something in the underlying NativeTheme has changed. This normally means that either the value of `shouldUseDarkColors`, `shouldUseHighContrastColors` or `shouldUseInvertedColorScheme` has changed. You will have to check them to determine which one has changed.

Properties

The `nativeTheme` module has the following properties:

`nativeTheme.shouldUseDarkColors` *Readonly*

A `boolean` for if the OS / Chromium currently has a dark mode enabled or is being instructed to show a dark-style UI. If you want to modify this value you should use `themeSource` below.

`nativeTheme.themeSource`

A `string` property that can be `system`, `light` or `dark`. It is used to override and supersede the value that Chromium has chosen to use internally.

Setting this property to `system` will remove the override and everything will be reset to the OS default. By default `themeSource` is `system`.

Settings this property to `dark` will have the following effects:

- `nativeTheme.shouldUseDarkColors` will be `true` when accessed
- Any UI Electron renders on Linux and Windows including context menus, devtools, etc. will use the dark UI.
- Any UI the OS renders on macOS including menus, window frames, etc. will use the dark UI.
- The `prefers-color-scheme` CSS query will match `dark` mode.
- The `updated` event will be emitted

Settings this property to `light` will have the following effects:

- `nativeTheme.shouldUseDarkColors` will be `false` when accessed
- Any UI Electron renders on Linux and Windows including context menus, devtools, etc. will use the light UI.

- Any UI the OS renders on macOS including menus, window frames, etc. will use the light UI.
- The `prefers-color-scheme` CSS query will match `light` mode.
- The `updated` event will be emitted

The usage of this property should align with a classic "dark mode" state machine in your application where the user has three options.

- `Follow OS` –> `themeSource = 'system'`
- `Dark Mode` –> `themeSource = 'dark'`
- `Light Mode` –> `themeSource = 'light'`

Your application should then always use `shouldUseDarkColors` to determine what CSS to apply.

`nativeTheme.shouldUseHighContrastColors` *macOS Windows Readonly*

A `boolean` for if the OS / Chromium currently has high-contrast mode enabled or is being instructed to show a high-contrast UI.

`nativeTheme.shouldUseInvertedColorScheme` *macOS Windows Readonly*

A `boolean` for if the OS / Chromium currently has an inverted color scheme or is being instructed to use an inverted color scheme.

`nativeTheme.inForcedColorsMode` *Windows Readonly*

A `boolean` indicating whether Chromium is in forced colors mode, controlled by system accessibility settings. Currently, Windows high contrast is the only system setting that triggers forced colors mode.