Building with palettes

Palettes are the lowest level system within UDS. They are simply a structured set of all the design tokens for a brand. Brand palettes enable you to build within the *constraints* of a brand's identity by using that brand's design tokens.

Using palettes in javascript projects

Brand palettes are published as npm packages following a `@telus-uds/palette-<brand>` naming convention. The published npm package contains web-specific and React Native-specific brand resources under the `/build/web` and `/build/rn` paths respectively. In the following examples you can substitute `rn` in for `web` to have the same effect.

Palette values

You can consume the palette values as a structured javascript object to use in your code as follows:

```
import palette from '@telus-uds/palette-koodo/build/web/palette.js'

const bgColor = palette.color.beeswax
```

Icons

To consume icons in a React application, simply import the icon as a react component.

```
import { Error as ErrorReactIcon } from '@telus-uds/palette-koodo/build/web/icons'
```

For advanced use cases, or non-React applications, the icon SVGs can also be imported.

```
// a raw svg - you'll need to provide a loader
import ErrorIcon from '@telus-uds/palette-
```

```
koodo/build/web/icons/error.svg'
```

Note: for React Native applications that are consuming the react native build of the palette, you will need to additionally install `react-native-svg` in order to render the react native icon components.

Fonts

Raw font resources are available via the brand palette - it is recommended to follow the guidance on font loading rather than access the resources directly.

Using palettes in CSS

INFO

coming soon!

Using palettes in native app projects

INFO

coming soon!

Edit this page