Develop a community palette

Brand palettes are collections of all the design tokens which represent the identity of a given brand. Each brand identity will map one-to-one to a brand palette. All teams building with UDS should use the appropriate brand palette(s) to ensure brand consistency. Learn more about brand palettes.

Universal Design System encourages the community to create palettes in the community space if there is a need to create a new palette for one of the products or sub-brands you are working on. This guide covers everything you need to know to `create a new palette` or `extend an existing palette`.

We have three `core palettes` that you could extend and create a palette from: `@telus-uds/palette-allium` is the base palette for TELUS, `@telus-uds/palette-koodo` for Koodo and `@telus-uds/palette-public-mobile` for Public Mobile.

> INFO
>
> - If this is the first time you are contributing to the Universal Design System repository
>   - You will need Github access
>   - You may need to request write access for you and your team
> - If you are contributing to the UDS project, please follow the instructions to develop for UDS

Using core palette vs extending palette vs creating new palette

Using core palette

You should use a `core palette` if you are working under one of the brands in the system (TELUS, Koodo or Public Mobile). Using the core palette of a given brand will help you to make sure that the tokens are aligned with the brand vision. This is the most common and straightforward use of palettes. Please refer to the brand orientation section of the guide for specific information.

If you are using `styled-components`, you will probably use values from a `core palette` on your stylesheet.

Extending palettes

> INFO
>
> Core palette extension should be used with discretion, please assess your use case with the UDS core team before creating a palette to avoid unnecessary palettes and have the contribution blocked downstream.

Extending palettes allows your team to add tokens to an existing palette or override the values on the palette. At this time, extension works only with `core palettes`.

1. If the token named on the extension **is not used** in the core palette, **that will generate a build file that contains tokens from the two palettes.** This means that any value on the core palette will still be updated alongside it and no token management effort is needed.

```
# core palette uses the following sizing scale
"size": {
    "values": {
      "size0": {
        "value": 0
      },
      "size1": {
        "value": 1
      }
    }
}

# Extended palette adds a new key-value pair
"size": {
    "values": {
      "size2": {
        "value": 2
    }
}

# The built palette should provide a merge between both
size: {
```

```
    size0: '0px',
    size1: '1px',
    size2: '2px'
    }
```

2. If the name of the token used on the extension **is used** in the core palette, **that will generate a build file that overrides the values** from the core palette with the value in the extension palette.

```
# core palette uses the following sizing scale
"size": {
    "values": {
      "size0": {
        "value": 0
      },
      "size1": {
        "value": 1
      }
    }
}

# Extended palette adds a new key-value pair
"size": {
    "values": {
      "size1": {
        "value": 10
    }
}

# The built palette should override the value from the core
palette
size: {
    size0: '0px',
    size1: '10px'
    }
```

Creating a new community palette

INFO

> Creating new palettes should be used with discretion, please assess your use case with the UDS core team before creating a palette to avoid unnecessary palettes and have the contribution blocked downstream.

If your team is going to work with a new brand, or if your product will not share the options from one of the existing brands that is a case fit for creating a new palette. New palettes are created from scratch and your team will have to define each and every value on it.

Scaffold your palette

We provide easy scripts to generate the basic scaffolding of a palette. From the root directory of the monorepo, run the following script and you'll be prompted to select a core palette if you choose to extend.

```
npm run create-palette
```

Each palette created or extended will live in the folder `packages/palettes/community`. The script will create the necessary files to get you started on a folder that concatenates the pattern palette and the name provide (e.g. If the name informed is UDS, the new folder will be called palette-UDS) to keep the consistency across the system.

Now you are all set to start adding entries to the `palette.json`. If you chose to extend an existing palette, the build process would create a merged `build/palette.js` with the original palette and assets merged from the base palette. So, basically, you get to override the design tokens that already exist or create new design tokens. If you want to use a different set of fonts or icons for your new / extended palette (or a subset of new fonts / icons) you can just create `font` and `icon` folders and add them in there. The `system-tokens` package implements the tooling that validates, builds and merges these palettes.

Test your palette

A palette will need a supporting theme for it to be applied and tested out on live components. You can follow the steps to develop a new theme to see the palette you've created in action.

Using community palettes

Community palettes are published as npm packages following a `@telus-uds/palettes-community.palette-<sub-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.

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/palettes-community.palette-casa/build/web/palette.js'

const bgColor = palette.color.casaPrimary
```

Icons

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

```
import { Error as ErrorReactIcon } from '@telus-uds/palettes-community.palette-casa/build/web/icons'
```

Fonts

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

Edit this page