Developer guide

Your entry point into UDS will be different, depending on how you intend to use UDS. If you're not sure how UDS fits in with your work you can always reach out to the UDS team on Slack.

Multi-brand web

For multi-brand web applications, we provide the `@telus-uds/components-web` package, which offers a comprehensive collection of web components. This package allows application teams to use any brand theme they desire. Unlike the branded packages like `@telus-uds/ds-allium` which is now deprecated, the `@telus-uds/components-web` package is specifically designed to be interoperable between different brands.

With `components-web`, the web components are multi-brand, meaning they can seamlessly adapt to various brand themes without compromising functionality or design. This approach offers a high level of flexibility, allowing app teams to easily make their applications work on different brands by simply switching the themes.

It is essential to highlight that the `components-web` package ensures consistent UI experiences across various brands while maintaining the ability to iterate and release components independently for each brand. By using the multi-brand web components, the app teams can streamline their development process and focus on creating a unified and engaging user experience that can be effortlessly extended to multiple brands.

Multi-brand multi-platform

If you are building a mobile application with Expo/React Native, or if you are developing a multi-platform application using React Native Web, `@telus-uds/components-base` is the essential package to use. It provides the necessary components and ensures compatibility across these platforms.

With `@telus-uds/components-base`, you can efficiently build and deploy applications for both mobile and web platforms, taking advantage of the multi-platform components' adaptability and flexibility. This enables your development team to create a unified and cohesive UI experience across all supported platforms.

Another important thing to remember is that in addition to the peer dependencies mentioned above, you'll also have to install `react-native-svg`:

```
yarn add react-native-svg
```

The rest is essentially the same as on web: all you need is to have your components wrapped in the appropriate provider.

Single-brand

For single brand applications, the development approach revolves around using the `@telus-uds/components-web` or `@telus-uds/components-base` package, depending on the specific platform you will be targeting. Both of these packages offer standardized components and variants that ensure consistency across different brands. These components can be seamlessly integrated into your application, providing a cohesive user experience.

In cases where a team requires non-standard variants to deliver a rich, brand-specific experience, we offer support for additional variants through brand-specific community packages. These additional variants are tailored to each brand's unique requirements, allowing teams to customize the components to align with their brand identity and style guidelines. However, it's crucial to note that these brand-specific additional variants are not cross-compatible and will only work within their respective brands. This ensures that each brand can maintain its distinct visual identity and user experience while still benefiting from the standardized components and variants provided by components-web or components-base.

Other technologies

If your application does not use React (or React Native) you can still build coherent branded experiences using UDS by leveraging the Palettes and Themes. We have decoupled the design decisions into independently installable packages, making it possible to create components in other languages that maintain the same visual styles. Follow our guide on building with Palettes.

Get support

- Join the #ds-support Slack channel for general discussions, announcements, and questions related to the design system.

Edit this page