Design System Components

When building components, the NEAR VM provides a complete set of [Radix primitives](#) to simplify UI development.

Radix UI

Using embedded Radix primitives on the NEAR VM is simple and straight-forward. You don't need to import any files:

return

(< Label . Root className = "LabelRoot"

     Hello

World ! < / Label . Root

) ; Limitations Currently, NEAR VM impose some limitations on the Radix UI framework:

- Form
- component is not available.
- You can't use.Portal
- definitions.
- Using CSS is different. You'll have to use astyled.div
- wrapper.

Using CSS

Here is an example on how to use CSS through thestyled.div wrapper:

const

Wrapper

= styled . div ` .SwitchRoot

{ ... } .SwitchThumb

{ ... } ` ;

return

(< Wrapper

     < Switch . Root className = "SwitchRoot"

     < Switch . Thumb className = "SwitchThumb"

/

     < / Switch . Root

     < / Wrapper

) ; Using Wrapper [Example widget using Wrapper](#)

Usingstyled-components

You can use[styled-components](#) in combination with Radix UI primitives. Here's an example:

const

SwitchRoot

=

styled ("Switch.Root") ` all : unset ; display : block ; width :

42 px ; height :

25 px ; background-color :

var (--blackA9) ; border-radius :

9999 px ; position : relative ; box-shadow :

0

2 px

10 px

var (--blackA7) ;

& [data-state = "checked"]

{ background-color :

black ; } ` ;

const

SwitchThumb

=

styled ("Switch.Thumb") ` all : unset ; display : block ; width :

21 px ; height :

21 px ; background-color :

white ; border-radius :

9999 px ; box-shadow :

0

2 px

2 px

var (--blackA7) ; transition : transform 100 ms ; transform :

translateX (2 px) ; will-change : transform ;

& [data-state = "checked"]

{ transform :

translateX (19 px) ; } ` ;

return

(< SwitchRoot

< SwitchThumb

/

< / SwitchRoot

) ; Using styled components [Example widget using styled components to style Radix UI](#).

Forward references

The NEAR VM re-implements [React's forwardRef](#) as ref="forwardedRef" .

You can use ref="forwardedRef" to forward references through to support Radix's asChild property:

Dialog.jsx < AlertDialog . Trigger asChild

< Widget src = "near/widget/TestButton" props = { {

label :

"Click Me"

} } /

      < / AlertDialog . Trigger

      TestButton.jsx const

Button

= styled . button ` background :

f00

; ` ;

return

(< Button type = "button" ref = "forwardedRef"

      { props . label } :

Forwarded < / Button

) ;

DIG components

These are the Design Interface Guidelines (DIG) components available on the NEAR VM:

- [DIG.Button](#)
- [DIG.Theme](#)

DIG.Button

A fully featured button component that can act as a

or tag.

DIG.Button properties [Click here](#) for properties and details.

DIG.Theme

This component wraps all of NEAR Components so you don't need to render it yourself.

tip You can use any of the [CSS variables](#) defined inside DIG.Theme . [Edit this page](#) Last updated on Apr 10, 2024 by gagdiez Was this page helpful? Yes No Need some help? [Chat with us](#) or check our [Dev Resources](#) ! [Twitter](#) [Telegram](#) [Discord](#) [Zulip](#)