

## Introducción

Empezando

## Componentes

Alert

Badge

Button

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

## Utilidades

useAction

useFormat

useInputMask

useKeyPress

useTable

useTheme

useTooltip

useValidator

## Composables

useFilters

useLogos

# Button

Documentación y ejemplos del componente `button`. Para definir **acciones en formularios, dialog o filtros**, con distintos colores, tamaños y estados.

## Props

### # text

Type: `String`

Define el **texto a mostrar** dentro del componente `button`.

Texto a mostrar

Texto a mostrar

```
<ButtonComponent
  data-eit-variant='blue'
  text="Texto a mostrar"
/>
```

html

### icon

Type: `String`

Define el **ícono a mostrar** dentro del componente `button`.

✓ Texto a mostrar

✓ Texto a mostrar

```
<ButtonComponent
  data-eit-variant='blue'
  text="Texto a mostrar"
  icon="fa-solid fa-circle-check"
/>
```

html

### iconClass

Type: `String`

## Introducción

### Empezando

## Componentes

Alert

Badge

Button

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

## Utilidades

useAction

useFormat

useInputMask

useKeyPress

useTable

useTheme

useTooltip

useValidator

## Composables

useFilters

useLogos

Texto a mostrar

```
<ButtonComponent
  data-eit-variant='blue'
  text="Texto a mostrar"
  icon="fa-solid fa-circle-check"
  iconClass="eit-color--green"
/>
```

html

## isDisabled

Type: `Boolean`

Esta propiedad **bloquea el botón** de los eventos clic del usuario.

Bloqueado

Bloqueado

```
<ButtonComponent
  data-eit-variant='blue'
  text="Bloqueado"
  isDisabled
/>
```

html

## loading

Type: `Boolean`

Esta propiedad define un estado `boolean` dentro del componente `button`.

```
<ButtonComponent
  data-eit-variant='blue'
  text="Texto a mostrar"
  :loading="loading"
/>
```

html

## loadingText

Type: `String`

Esta propiedad define el texto que se muestra en el estado `loading`

```
<ButtonComponent
  data-eit-variant='blue'
  text="Texto a mostrar"
```

html

## Introducción

Empezando

## Componentes

Alert

Badge

Button

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

## Utilidades

useAction

useFormat

useInputMask

useKeyPress

useTable

useTheme

useTooltip

useValidator

## Composables

useFilters

useLogos

# Emits

## emitEvent

Función que notifica **cuando se presiona** el componente `button`. Se debe usar un manejador en el componente padre que reciba esta información.

```
<script setup lang="ts">
  const handleEmit = () => {
    //Aquí va el código a ejecutarse desde la acción.
  }
</script>

<template>
  <ButtonComponent
    data-eit-variant='blue'
    text="Botón con emit"
    @emitEvent="handleEmit"
  />
</template>
```

vue

# Styles

## colors

**Solid** → Se puede definir con la clase `eit-btn` y su modificador `--color` o con su `data-attributes` respectivo.

class	data-attributes
<code>.eit-btn--primary</code>	<code>data-eit-variant="primary"</code>
<code>.eit-btn--secondary</code>	<code>data-eit-variant="secondary"</code>
<code>.eit-btn--tertiary</code>	<code>data-eit-variant="tertiary"</code>
<code>.eit-btn--green</code>	<code>data-eit-variant="green"</code>
<code>.eit-btn--red</code>	<code>data-eit-variant="red"</code>
<code>.eit-btn--yellow</code>	<code>data-eit-variant="yellow"</code>

Introducción

Empezando

Componentes

Alert

Badge

Button

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

Utilidades

useAction

useFormat

useInputMask

useKeyPress

useTable

useTheme

useTooltip

useValidator

Composables

useFilters

useLogos

.eit-btn--gray

data-eit-variant="gray"

--primary

--secondary

--tertiary

--green

--red

--yellow

--blue

--gray

Ejemplo

```
<!--class-->
<ButtonComponent
  class='eit-btn--primary'
  text="--primary"
/>

<!--data-attributes-->
<ButtonComponent
  data-eit-variant='primary'
  text="--primary"
/>
```

html

Outline → Se puede definir con la clase `eit-btn-outline` y su modificador `--color` o con su `data-attributes` respectivo.

class	data-attributes
<code>.eit-btn-outline--primary</code>	<code>data-eit-variant="primary"</code> <code>data-eit-outline</code>
<code>.eit-btn-outline--secondary</code>	<code>data-eit-variant="secondary"</code> <code>data-eit-outline</code>
<code>.eit-btn-outline--tertiary</code>	<code>data-eit-variant="tertiary"</code> <code>data-eit-outline</code>
<code>.eit-btn-outline--green</code>	<code>data-eit-variant="green"</code> <code>data-eit-outline</code>
<code>.eit-btn-outline--red</code>	<code>data-eit-variant="red"</code> <code>data-eit-outline</code>
<code>.eit-btn-outline--yellow</code>	<code>data-eit-variant="yellow"</code> <code>data-eit-outline</code>
<code>.eit-btn-outline--blue</code>	<code>data-eit-variant="blue"</code> <code>data-eit-outline</code>

Introducción

Empezando

Componentes

Alert

Badge

Button

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

Utilidades

useAction

useFormat

useInputMask

useKeyPress

useTable

useTheme

useTooltip

useValidator

Composables

useFilters

useLogos

.eit-btn-outline--gray

data-eit-outline

--primary

--secondary

--tertiary

--green

--red

--yellow

--blue

--gray

Ejemplo

```
<!--class-->
<ButtonComponent
  class='eit-btn-outline--primary'
  text="--primary"
/>

<!--data-attributes-->
<ButtonComponent
  data-eit-variant='primary'
  data-eit-outline
  text="--primary"
/>
```

html

Small → Se define con la clase `eit-btn` y su modificador `--small` o con su `data-attributes` respectivo.

class	data-attributes
<code>.eit-btn--small</code>	<code>data-eit-size="small"</code>

--small

--small

Ejemplo

```
<!--class-->
<ButtonComponent
  class='eit-btn--primary eit-btn--small'
  text="--primary"
/>

<!--data-attributes-->
<ButtonComponent
  data-eit-variant='blue'
  data-eit-size='small'
  text="Texto a mostrar"
/>
```

html

## Introducción

Empezando

## Componentes

Alert

Badge

Button

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

## Utilidades

useAction

useFormat

useInputMask

useKeyPress

useTable

useTheme

useTooltip

useValidator

## Composables

useFilters

useLogos

class	data-attributes
<code>.eit-btn--square</code>	<code>data-eit-shape="square"</code>
<code>.eit-btn--square-small</code>	<code>data-eit-shape="square-small"</code>



## Ejemplo

```
html
<!--class-->
<ButtonComponent
  class='eit-btn--primary eit-btn--square'
  text="B"
/>

<!--data-attributes-->
<ButtonComponent
  data-eit-variant='blue'
  data-eit-shape='square-small'
  data-eit-outline
  text="B"
/>
```

## Configurar loading

Para configurar debes combinar las propiedades `loading` , `loadingText` y declarar el `@emitEvent` con una función controladora.



## Ejemplo

```
vue
<script setup lang="ts">
  import { ref } from 'vue'

  const loading = ref(false)

  const handleLoading = () => {
    loading.value = true
    setTimeout(() => loading.value = false, "1000")
  }
</script>

<template>
```

## Introducción

Empezando

## Componentes

Alert

Badge

**Button**

Detail

Dialog

Dropdown

Errors

Filters

Input

Loading

Pagination

ProgressBar

Select

Tab

Table

TextareaResize

Toast

## Utilidades

useAction

useFormat

useInputMask

useKeypress

useTable

useTheme

useTooltip

useValidator

## Composables

useFilters

useLogos

```

text="Loading"
:loading="loading"
loadingText="Guardando..."
@emitEvent="handleLoading"
/>
</template>

```

Previous page

[Badge](#)

Next page

[Detail](#)