

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

Dialog

Documentación y ejemplos del componente `dialog`. Para incorporar **cuadros de diálogo**, con formularios cortos o notificaciones de usuario.

Props

loading

Type: `Boolean`

Define un **elemento de carga** antes de mostrar el contenido del `dialog`

`dialog loading`

```
<DialogComponent
  ref="dialogLoading"
  data-eit-position='top'
  :loading="loadingDialog"
>
  <template #head>
    <h3
      data-eit-font-size="x5"
      data-eit-my="0"
    >
      Dialog loading
    </h3>
  </template>
  <template #content>
    Texto simple de dialog loading.
  </template>
</DialogComponent>
```

html

btnSubmit

Type: `Boolean`

Habilita el **botón submit** dentro del componente `dialog`

`dialog submit`

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

```
data-eit-position="top"
:btnSubmit="true"
>
<template #head>
  <h3
    data-eit-font-size="x5"
    data-eit-my="0"
  >
    Dialog submit
  </h3>
</template>
<template #content>
  Texto simple de dialog submit.
</template>
</DialogComponent>
```

loadingSubmit

Type: **Boolean**

Para agregar un estado **loading** al botón submit. Debes definir una constante **boolean** para manejar el estado, más información de la implemetación en el componente [button](#)

dialog loading submit

```
<DialogComponent
  ref="dialogLoadingSubmit"
  data-eit-position="top"
  :btnSubmit="true"
  :loadingSubmit="loadingDialogSubmit"
>
  <template #head>
    <h3
      data-eit-font-size="x5"
      data-eit-my="0"
    >
      Dialog loading submit
    </h3>
  </template>
  <template #content>
    Texto simple de dialog loading submit.
  </template>
</DialogComponent>
```

html

disabledSubmit

Type: **Boolean**

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

dialog disabled submit

```
<DialogComponent
  ref="dialogDisabledSubmit"
  data-eit-position="top"
  :btnSubmit="true"
  :disabledSubmit="disabledSubmitDialog"
>
<template #head>
  <h3
    data-eit-font-size="x5"
    data-eit-my="0"
  >
    Dialog disabled submit
  </h3>
</template>
<template #content>
  Texto simple de dialog disabled submit.
</template>
</DialogComponent>
```

html

btnSubmitConfig

Type: `Boolean`

Puedes **personalizar el botón** submit creando un objeto que revisa la propiedad.

```
{ text: String, loadingText: String, variant: String }
```

dialog config submit

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

  const configSubmitDialog = ref({
    text: 'Personalizado',
    loadingText: 'Personalizando...',
    variant: 'green'
  })
</script>

<template>
  <DialogComponent
    ref="dialogConfigSubmit"
    data-eit-position="top"
    :btnSubmit="true"
    :btnSubmitConfig="configSubmitDialog"
  >
```

vue

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

```

        data-eit-font-size="x5"
        data-eit-my="0"
      >
        Dialog config submit
      </h3>
    </template>
    <template #content>
      Texto simple de dialog config submit.
    </template>
  </DialogComponent>
</template>

```

Emits

emitSubmit

Función que notifica cuando se **presiona el botón submit** dentro del componente. Se debe usar un manejador en el componente padre que reciba esta información.

```

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

<template>
  <DialogComponent
    data-eit-position="top"
    :btnSubmit="true"
    @emitSubmit="handleEmitSubmit"
  >
    <template #head>
      <h3
        data-eit-font-size="x5"
        data-eit-my="0"
      >
        Dialog emit submit
      </h3>
    </template>
    <template #content>
      Texto simple de dialog emit submit.
    </template>
  </DialogComponent>
</template>

```

vue

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

componente. Se debe usar un manejador en el componente padre que reciba esta información.

<script setup lang="ts">

const handleEmitClose = () => {

//Aquí va el código a ejecutarse desde la acción.

}

</script>

<template>

<DialogComponent

@emitCloseDialog="handleEmitClose"

>

<template #head>

<h3

>

Dialog emit close

</h3>

</template>

<template #content>

Texto simple de dialog emit close.

</template>

</DialogComponent>

</template>

Styles

Size

Para definir el ancho que tendrá el componente, puedes usar `class` o `data-attributes` .

class	data-attributes
.eit-dialog--small	data-eit-size="small"
.eit-dialog--large	data-eit-size="large"
.eit-dialog--fullscreen	data-eit-size="fullscreen"

dialog

dialog small

dialog large

dialog fullscreen

Ejemplo

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

```
data-eit-size='large'
>
<template #head>
  <h3
    data-eit-font-size="x5"
    data-eit-my="0"
  >
    Dialog
  </h3>
</template>
<template #content>
  Texto simple de dialog.
</template>
</DialogComponent>
```

Position

Para definir la **posición** que tendrá el componente, puedes usar `class` o `data-attributes`.

class	data-attributes
<code>.eit-dialog--top</code>	<code>data-eit-position="top"</code>

dialog top

Ejemplo

```
<DialogComponent
  ref="dialogTop"
  data-eit-position='top'
>
  <template #head>
    <h3
      data-eit-font-size="x5"
      data-eit-my="0"
    >
      Dialog
    </h3>
  </template>
  <template #content>
    Texto simple de dialog.
  </template>
</DialogComponent>
```

html

Scrollable

- 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
- useMessage
- useRequired
- useRules

Styles

- Background

class	data-attributes
<code>.eit-dialog--scrollable</code>	<code>data-eit-scroll="true"</code>

dialog scroll

Ejemplo

html

```
<DialogComponent
  ref="dialogTop"
  data-eit-scroll='true'
>
  <template #head>
    <h3
      data-eit-font-size="x5"
      data-eit-my="0"
    >
      Dialog
    </h3>
  </template>
  <template #content>
    Texto simple de dialog.
  </template>
</DialogComponent>
```