Message

Utilizado para mostrar retroalimentación después de una actividad. La diferencia con el componente Notification es que este ultimo es utilizado para mostrar una notificación pasiva a nivel de sistema.

Uso básico

Se muestra en la parte superior de la pagina y desaparece después de 3 segundos.

:::demo La configuración del componente Message es muy similar al del componente notification, así que parte de las opciones no serán explicadas en detalle aquí. Puedes consultar la tabla de opciones en la parte inferior combinada con la documentación del componente notification para comprenderla. Element a registrado un método `$message` para poder invocarlo. Message puede tomar una cadena o un Vnode como parámetro, y lo mostrara como el cuerpo principal.

```
<template>
  <el-button :plain="true" @click="open">Show message</el-button>
  <el-button :plain="true" @click="openVn">VNode</el-button>
</template>

<script>
  export default {
    methods: {
      open() {
        this.$message('This is a message.');
      },

      openVn() {
        const h = this.$createElement;
        this.$message({
          message: h('p', null, [
            h('span', null, 'Message can be '),
            h('i', { style: 'color: teal' }, 'VNode')
          ])
        });
      }
    }
  }
</script>
```

:::

Tipos

Utilizados para mostrar retroalimentación de Success, Warning, Message y Error activities.

:::demo Cuando necesite mas personalización, el componente Message también puede tomar un objeto como parámetro. Por ejemplo, estableciendo el valor de `type` puede definir diferentes tipos, el predeterminado es `info`. En tales casos el cuerpo principal se pasa como el valor de `message`. También, hay registrados métodos para los diferentes tipos, así que, puedes llamarlos sin necesidad de pasar un tipo como `open4`.

```
<template>
  <el-button :plain="true" @click="open2">success</el-button>
  <el-button :plain="true" @click="open3">warning</el-button>
  <el-button :plain="true" @click="open1">message</el-button>
  <el-button :plain="true" @click="open4">error</el-button>
</template>

<script>
  export default {
    methods: {
      open1() {
        this.$message('This is a message.');
      },
      open2() {
        this.$message({
          message: 'Congrats, this is a success message.',
          type: 'success'
        });
      },

      open3() {
        this.$message({
          message: 'Warning, this is a warning message.',
          type: 'warning'
        });
      },

      open4() {
        this.$message.error('Oops, this is a error message.');
      }
    }
  }
</script>
```

:::

Closable

Un botón para cerrar que puede ser agregado.

:::demo Un componente Message predeterminado no se puede cerrar manualmente. Si necesitas un componente message que pueda cerrarse, puedes establecer el campo `showClose`. Ademas, al igual que las notificaciones, message tiene un atributo `duration` que puede ser controlado. Por defecto la duración es de 3000 ms, y no desaparecerá al llegar a `0`.

```
<template>
  <el-button :plain="true" @click="open1">message</el-button>
  <el-button :plain="true" @click="open2">success</el-button>
  <el-button :plain="true" @click="open3">warning</el-button>
  <el-button :plain="true" @click="open4">error</el-button>
</template>

<script>
  export default {
    methods: {
      open1() {
        this.$message({
          showClose: true,
          message: 'This is a message.'
        });
      },

      open2() {
        this.$message({
          showClose: true,
          message: 'Congrats, this is a success message.',
          type: 'success'
        });
      },

      open3() {
        this.$message({
          showClose: true,
          message: 'Warning, this is a warning message.',
          type: 'warning'
        });
      },

      open4() {
        this.$message({
          showClose: true,
          message: 'Oops, this is a error message.',
          type: 'error'
        });
      }
```

```
    }
  }
</script>
```

:::

Texto centrado

Utiliza el atributo `center` para centrar el texto.

```
<template>
  <el-button :plain="true" @click="openCenter">Centered text</el-button>
</template>

<script>
  export default {
    methods: {
      openCenter() {
        this.$message({
          message: 'Centered text',
          center: true
        });
      }
    }
  }
</script>
```

Utiliza cadenas HTML

`message` soporta cadenas HTML.

:::demo Establece la propiedad `dangerouslyUseHTMLString` en true y `message` sera tratado como una cadena HTML.

```
<template>
  <el-button :plain="true" @click="openHTML">Use HTML String</el-button>
</template>

<script>
  export default {
    methods: {
      openHTML() {
        this.$message({
          dangerouslyUseHTMLString: true,
          message: '<strong>This is <i>HTML</i> string</strong>'
        });
      }
    }
```

```
  }
</script>
```

:::

Aunque la propiedad `message` soporta cadenas HTML, realizar arbitrariamente render dinámico de HTML en nuestro sitio web puede ser muy peligroso ya que puede conducir fácilmente a XSS attacks. Entonces cuando `dangerouslyUseHTMLString` esta activada, asegúrese que el contenido de `message` sea de confianza, y **nunca** asignar `message` a contenido generado por el usuario.

Métodos Globales

Element ha agregado un método global llamado `$message` para Vue.prototype. Entonces en una instancia de vue puede llamar a `Message` como lo hicimos en esta pagina.

Importación local

Import `Message`:

```
import { Message } from 'element-ui';
```

En este caso debería llamar al método `Message(options)`. También se han registrado métodos para los diferentes tipos, e.g. `Message.success(options)`. Puede llamar al método `Message.closeAll()` para cerrar manualmente todas las instancias.

Opciones

Atributo	Descripción	Tipo	Valores permitidos	Por defecto
message	texto del mensaje	string / VNode	—	—
type	tipo del mensaje	string	success/warning/info/error	info
iconClass	clase personalizada para el icono, sobrescribe `type`	string	—	—
dangerouslyUseHTMLString	Habilitar para que `message` sea tratado como cadena HTML	boolean	—	false
customClass	nombre de clase personalizado para el componente Message	string	—	—

Atributo	Descripción	Tipo	Valores permitidos	Por defecto
duration	muestra la duración,en mili segundos. si se establece en 0, este no se apagara automáticamente	number	—	3000
showClose	utilizado para mostrar un botón para cerrar	boolean	—	false
center	utilizado para centrar el texto	boolean	—	false
onClose	función callback ejecutada cuando se cierra con una instancia de mensaje como parámetro	function	—	—
offset	La distancia desde la parte superior del viewport	number	—	20

Métodos

`Message` y `this.$message` regresan una instancia del componente Message. Para cerrar manualmente la instancia, puede llamar al método `close`.

Método	Descripción
close	cierra el componente Message