Notification

Muestra un mensaje de notificación global en una esquina de la página.

Uso básico

:::demo Element ha registrado el método`$notify` y recibe un objeto como parámetro. En el caso más sencillo, puede establecer el campo de `title` y el campo de `message` para el título y el cuerpo de la notificación. De forma predeterminada, la notificación se cierra automáticamente después de 4500ms, pero configurando `duration` se puede controlar su duración. Específicamente, si está configurado en `0`, no se cerrará automáticamente. Tenga en cuenta que `duration` recibe un `Number` en mili segundos.

```
<template>
  <el-button
    plain
    @click="open1">
    Closes automatically
  </el-button>
  <el-button
    plain
    @click="open2">
    Won't close automatically
    </el-button>
</template>

<script>
  export default {
    methods: {
      open1() {
        const h = this.$createElement;

        this.$notify({
          title: 'Title',
          message: h('i', { style: 'color: teal' }, 'This is a reminder')
        });
      },

      open2() {
        this.$notify({
          title: 'Prompt',
          message: 'This is a message that does not automatically close',
          duration: 0
        });
      }
    }
```

```
  }
</script>
```

:::

Tipos de notificaciones

Proporcionamos cuatro tipos: success, warning, info y error.

:::demo Element proporciona cuatro tipos de notificación: `success`, `warning`, `info` y `error`. Se definen por el campo `type` y se ignorarán otros valores. También se han registrado métodos para estos tipos que se pueden invocar directamente como en el ejemplo `open3` y `open4` sin pasar un campo `type`.

```
<template>
  <el-button
    plain
    @click="open1">
    Success
  </el-button>
  <el-button
    plain
    @click="open2">
    Warning
  </el-button>
  <el-button
    plain
    @click="open3">
    Info
  </el-button>
  <el-button
    plain
    @click="open4">
    Error
  </el-button>
</template>

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

```
      open2() {
        this.$notify({
          title: 'Warning',
          message: 'This is a warning message',
          type: 'warning'
        });
      },

      open3() {
        this.$notify.info({
          title: 'Info',
          message: 'This is an info message'
        });
      },

      open4() {
        this.$notify.error({
          title: 'Error',
          message: 'This is an error message'
        });
      }
    }
  }
</script>
```

:::

Posición personalizada

La notificación puede surgir de cualquier rincón que uno desee.

:::demo El atributo `position` define desde qué esquina se desliza la notificación. Puede ser `top-right`, `top-left`, `bottom-right` o `bottom-left`. Predeterminado: `top-right`.

```
<template>
  <el-button
    plain
    @click="open1">
    Top Right
  </el-button>
  <el-button
    plain
    @click="open2">
    Bottom Right
  </el-button>
```

```
  <el-button
    plain
    @click="open3">
    Bottom Left
  </el-button>
  <el-button
    plain
    @click="open4">
    Top Left
  </el-button>
</template>

<script>
  export default {
    methods: {
      open1() {
        this.$notify({
          title: 'Custom Position',
          message: 'I\'m at the top right corner'
        });
      },

      open2() {
        this.$notify({
          title: 'Custom Position',
          message: 'I\'m at the bottom right corner',
          position: 'bottom-right'
        });
      },

      open3() {
        this.$notify({
          title: 'Custom Position',
          message: 'I\'m at the bottom left corner',
          position: 'bottom-left'
        });
      },

      open4() {
        this.$notify({
          title: 'Custom Position',
          message: 'I\'m at the top left corner',
          position: 'top-left'
        });
      }
    }
```

```
  }
</script>
```

:::

Desplazamiento

Personalizar el desplazamiento de notificación desde el borde de la pantalla.

:::demo Configure el atributo `offset` para personalizar el desplazamiento de la notificación desde el borde de la pantalla. Tenga en cuenta que cada instancia de la notificación del mismo momento debe tener el mismo desplazamiento.

```
<template>
  <el-button
    plain
    @click="open">
    Notification with offset
  </el-button>
</template>

<script>
  export default {
    methods: {
      open() {
        this.$notify.success({
          title: 'Success',
          message: 'This is a success message',
          offset: 100
        });
      }
    }
  }
</script>
```

:::

Usando cadenas HTML

`message` soporta cadenas HTML.

:::demo Configure `dangerouslyUseHTMLString` a true y `message` se tratará como una cadena HTML.

```
<template>
  <el-button
    plain
    @click="open">
    Use HTML String
```

```
  </el-button>
</template>

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

:::

Aunque la propiedad `message` soporta cadenas HTML, el renderizado dinámico de HTML arbitrario en su sitio web puede ser muy peligroso porque puede conducir fácilmente a ataques XSS. Por lo tanto, cuando `dangerouslyUseHTMLString` está a true, por favor asegúrese de que el contenido del mensaje es confiable, y **nunca** asigne `message` al contenido proporcionado por el usuario.

Ocultar boton de cerrar

Es posible ocultar el botón de cerrar

:::demo Configure el atributo `showClose` como `false` para que el usuario no pueda cerrar la notificación.

```
<template>
  <el-button
    plain
    @click="open">
    Hide close button
    </el-button>
</template>

<script>
  export default {
    methods: {
      open() {
        this.$notify.success({
          title: 'Info',
          message: 'This is a message without close button',
          showClose: false
```

```
        });
      }
    }
  }
</script>
```

:::

Método global

Element ha añadido un método global `$notify` para Vue.prototype. Así que en una instancia de vue se puede llamar `Notification` como lo hacemos en esta página.

Importar localmente

Importar `Notification`:

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

En este caso, debe llamar a `Notification(options)`. También se han registrado métodos para diferentes tipos, e.j. `Notification.success(options)`. Puede llamar al método `Notification.closeAll()` para cerrar manualmente todas las instancias.

Opciones

Atributo	Descripción	Tipo	Valores aceptados	Por defecto
title	titulo	string	—	—
message	mensaje	string/Vue.VNode	—	—
dangerouslyUseHTMLString	si message es tratado como una cadena HTML	boolean	—	false
type	tipo de notificación	string	success/warning/info/error	
iconClass	clase personalizada de icono. Será anulado por `type`	string	—	—
customClass	nombre de clase personalizado para la notificación	string	—	—
duration	duración antes de cerrar. Si no se quiere que se cierre automáticamente este valor debe estar a 0	number	—	4500

Atributo	Descripción	Tipo	Valores aceptados	Por defecto
position	posición personalizada	string	top-right/top-left/bottom-right/bottom-left	top-right
showClose	si se muestra el botón de cerrar	boolean	—	true
onClose	función que se ejecuta cuando la notificación se cierra	function	—	—
onClick	función que se ejecuta cuando se hace clic en la notificación	function	—	—
offset	desplazamiento desde el borde superior de la pantalla. Cada instancia de notificación del mismo momento debe tener siempre el mismo desplazamiento.	number	—	0

Métodos

`Notification` y `this.$notify` devuelven la instancia de la notificación actual. Para cerrar manualmente la instancia, se puede llamar `close` para ello.

Método	Descripción
close	cierra la notificación