Loading

Show animation while loading data.

Loading inside a container

Displays animation in a container (such as a table) while loading data.

:::demo Element provides two ways to invoke Loading: directive and service. For the custom directive `v-loading`, you just need to bind a `boolean` value to it. By default, the loading mask will append to the element where the directive is used. Adding the `body` modifier makes the mask append to the body element.

```
<template>
  <el-table
    v-loading="loading"
    :data="tableData"
    style="width: 100%">
    <el-table-column
      prop="date"
      label="Date"
      width="180">
    </el-table-column>
    <el-table-column
      prop="name"
      label="Name"
      width="180">
    </el-table-column>
    <el-table-column
      prop="address"
      label="Address">
    </el-table-column>
  </el-table>
</template>

<style>
  body {
    margin: 0;
  }
</style>

<script>
  export default {
    data() {
      return {
        tableData: [{
          date: '2016-05-02',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }, {
          date: '2016-05-04',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }, {
          date: '2016-05-01',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }],
        loading: true
      };
    }
  };
</script>
```

:::

Customization

You can customize loading text, loading spinner and background color.

:::demo Add attribute `element-loading-text` to the element on which `v-loading` is bound, and its value will be displayed under the spinner. Similarly, `element-loading-spinner` and `element-loading-background` are for customizing loading spinner class name and background color.

```
<template>
  <el-table
    v-loading="loading"
    element-loading-text="Loading..."
    element-loading-spinner="el-icon-loading"
    element-loading-background="rgba(0, 0, 0, 0.8)"
    :data="tableData"
    style="width: 100%">
    <el-table-column
      prop="date"
      label="Date"
      width="180">
    </el-table-column>
    <el-table-column
      prop="name"
      label="Name"
      width="180">
    </el-table-column>
    <el-table-column
```

```
      prop="address"
      label="Address">
    </el-table-column>
  </el-table>
</template>

<script>
  export default {
    data() {
      return {
        tableData: [{
          date: '2016-05-02',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }, {
          date: '2016-05-04',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }, {
          date: '2016-05-01',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }],
        loading: true
      };
    }
  };
</script>
```

:::

Full screen loading

Show a full screen animation while loading data.

:::demo When used as a directive, a full screen Loading requires the `fullscreen` modifier, and it will be appended to body. In this case, if you wish to disable scrolling on body, you can add another modifier `lock`. When used as a service, Loading will be full screen by default.

```
<template>
  <el-button
    type="primary"
    @click="openFullScreen1"
    v-loading.fullscreen.lock="fullscreenLoading">
    As a directive
  </el-button>
  <el-button
    type="primary"
    @click="openFullScreen2">
    As a service
  </el-button>
</template>

<script>
  export default {
    data() {
      return {
        fullscreenLoading: false
      }
    },
    methods: {
      openFullScreen1() {
        this.fullscreenLoading = true;
        setTimeout(() => {
          this.fullscreenLoading = false;
        }, 2000);
      },
      openFullScreen2() {
        const loading = this.$loading({
          lock: true,
          text: 'Loading',
          spinner: 'el-icon-loading',
          background: 'rgba(0, 0, 0, 0.7)'
        });
        setTimeout(() => {
          loading.close();
        }, 2000);
      }
    }
  }
</script>
```

:::

Service

You can also invoke Loading with a service. Import Loading service:

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

Invoke it:

```
Loading.service(options);
```

The parameter `options` is the configuration of Loading, and its details can be found in the following table. `LoadingService` returns a Loading instance, and you can close it by invoking its `close` method:

```
let loadingInstance = Loading.service(options);
this.$nextTick(() => { // Loading should be closed asynchronously
  loadingInstance.close();
});
```

Note that in this case the full screen Loading is singleton. If a new full screen Loading is invoked before an existing one is closed, the existing full screen Loading instance will be returned instead of actually creating another Loading instance:

```
let loadingInstance1 = Loading.service({ fullscreen: true });
let loadingInstance2 = Loading.service({ fullscreen: true });
console.log(loadingInstance1 === loadingInstance2); // true
```

Calling the `close` method on any one of them can close this full screen Loading.

If Element is imported entirely, a globally method `$loading` will be registered to Vue.prototype. You can invoke it like this: `this.$loading(options)`, and it also returns a Loading instance.

Options

Attribute	Description	Type	Accepted Values	Default
target	the DOM node Loading needs to cover. Accepts a DOM object or a string. If it's a string, it will be passed to `document.querySelector` to get the corresponding DOM node	object/string	—	document.body
body	same as the `body` modifier of `v-loading`	boolean	—	false

Attribute	Description	Type	Accepted Values	Default
fullscreen	same as the `fullscreen` modifier of `v-loading`	boolean	—	true
lock	same as the `lock` modifier of `v-loading`	boolean	—	false
text	loading text that displays under the spinner	string	—	—
spinner	class name of the custom spinner	string	—	—
background	background color of the mask	string	—	—
customClass	custom class name for Loading	string	—	—