

Loading

Affiche une animation durant le chargement de données.

Loading dans un conteneur

Affiche une animation dans un conteneur (Comme un tableau par exemple) pendant le chargement des données.

Vue Element fournit deux moyens d'invoquer Loading: la directive et le service. Pour la directive `v-loading`, attachez simplement un `boolean`. Par défaut le masque sera ajouté à l'élément contenant la directive. Ajoutez le modificateur `body` pour ajouter le masque à l'élément body.

```
<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="Nom"
      width="180">
    </el-table-column>
    <el-table-column
      prop="address"
      label="Adresse">
    </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>

```

...

Personnalisation

Vous pouvez personnaliser le texte, le spinner et la couleur de fond.

:::demo Ajoutez l'attribut `element-loading-text` à l'élément sur lequel `v-loading` est attaché et sa valeur sera affichée sous le spinner. De la même façon, `element-loading-spinner` et `element-loading-background` permettent de personnaliser le spinner et la couleur de fond.

```

<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="Nom"
      width="180">
    </el-table-column>
    <el-table-column
      prop="address"
      label="Adresse">
    </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>

```

...

Chargement plein écran

Affichez une animation en plein écran quand vous chargez des données.

:::demo Quand il est utilisé comme une directive, un Loading plein écran nécessite le modificateur `fullscreen` qui sera ajouté au body. Dans ce cas, si vous souhaitez désactiver le défilement du body, vous pouvez ajouter le modificateur `lock`. Quand il est utilisé comme service, Loading est en plein écran par défaut.

```

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

Vous pouvez invoquer Loading comme un service. Importez le service Loading:

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

Et invoquer-le:

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

Le paramètre `options` correspond à la configuration de Loading (voir table suivante). `LoadingService` retourne une instance de Loading, que vous pouvez fermer en appelant la méthode `close` :

```

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

```

Notez que dans ce cas le Loading plein écran est un singleton. Si un nouveau Loading plein écran est invoqué avant la fermeture du précédent, celui-ci sera retourné au lieu d'en créer un nouveau:

```

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

```

Appeler la méthode `close` sur n'importe lequel des deux fermera le Loading.

Si Element est importé en entier, une méthode globale `$loading` sera ajoutée à `Vue.prototype`. Vous pourrez l'invoquer comme ceci: `this.$loading(options)` et elle retournera une instance Loading.

Options

Attribut	Description	Type	Valeurs acceptées	Défaut
target	L'élément du DOM a couvrir. Accepte un objet DOM ou un string. Si c'est un string, il sera passé à <code>document.querySelector</code> Pour avoir l'élément du DOM correspondant.	object/string	—	document.body
body	Identique au modificateur <code>body</code> de <code>v-loading</code> .	boolean	—	false
fullscreen	Identique au modificateur <code>fullscreen</code> de <code>v-loading</code> .	boolean	—	true
lock	Identique au modificateur <code>lock</code> de <code>v-loading</code> .	boolean	—	false
text	Texte a afficher sous le spinner.	string	—	—
spinner	Classe du spinner.	string	—	—
background	Couleur de fond du masque.	string	—	—
customClass	Classe du Loading.	string	—	—