Dialog

Informe les utilisateurs tout en préservant l'état de la page.

Usage

Le Dialog ouvre un modal personnalisable.

:::demo Configurez l'attribut `visible` avec un `Boolean`, un modal apparaîtra quand la valeur sera à `true`. Le Dialog possède deux parties: `body` et `footer`, ce-dernier nécessitant un `slot` appelé `footer`. L'attribut optionnel `title` (vide par défaut) définit le titre. Cet exemple montre également comment `before-close` peut être utilisé.

```
<el-button type="text" @click="dialogVisible = true">Cliquez pour ouvrir le
modal</el-button>

<el-dialog
  title="Tips"
  :visible.sync="dialogVisible"
  width="30%"
  :before-close="handleClose">
  <span>Ceci est un message</span>
  <span slot="footer" class="dialog-footer">
    <el-button @click="dialogVisible = false">Annuler</el-button>
    <el-button type="primary" @click="dialogVisible = false">Confirmer</el-button>
  </span>
</el-dialog>

<script>
  export default {
    data() {
      return {
        dialogVisible: false
      };
    },
    methods: {
      handleClose(done) {
        this.$confirm('Voulez-vous vraiment quitter ?')
          .then(_ => {
            done();
          })
          .catch(_ => {});
      }
    }
  };
</script>
```

:::

:::tip `before-close` ne fonctionne que quand l'utilisateur clique sur l'icône de fermeture en dehors du modal. S'il y a des boutons dans le `footer`, vous pouvez configurer `before-close` grâce à leur évènement click. :::

Personalisation

Le contenu du modal peut être n'importe quoi, tableau ou formulaire compris.

:::demo

```
<!-- Table -->
<el-button type="text" @click="dialogTableVisible = true">Ouvrir un modal avec
tableau</el-button>

<el-dialog title="Adresse d'expédition" :visible.sync="dialogTableVisible">
  <el-table :data="gridData">
    <el-table-column property="date" label="Date" width="150"></el-table-column>
    <el-table-column property="name" label="Nom" width="200"></el-table-column>
    <el-table-column property="address" label="Adresse"></el-table-column>
  </el-table>
</el-dialog>

<!-- Form -->
<el-button type="text" @click="dialogFormVisible = true">Ouvrir un modal avec
formulaire</el-button>

<el-dialog title="Adresse d'expédition" :visible.sync="dialogFormVisible">
  <el-form :model="form">
    <el-form-item label="Nom de promotion" :label-width="formLabelWidth">
      <el-input v-model="form.name" autocomplete="off"></el-input>
    </el-form-item>
    <el-form-item label="Zones" :label-width="formLabelWidth">
      <el-select v-model="form.region" placeholder="Sélectionnez une zone">
        <el-option label="Zone No.1" value="shanghai"></el-option>
        <el-option label="Zone No.2" value="beijing"></el-option>
      </el-select>
    </el-form-item>
  </el-form>
  <span slot="footer" class="dialog-footer">
    <el-button @click="dialogFormVisible = false">Annuler</el-button>
    <el-button type="primary" @click="dialogFormVisible = false">Confirmer</el-
button>
  </span>
</el-dialog>

<script>
  export default {
    data() {
      return {
        gridData: [{
          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'
        }, {
          date: '2016-05-03',
          name: 'John Smith',
          address: 'No.1518,  Jinshajiang Road, Putuo District'
        }],
        dialogTableVisible: false,
        dialogFormVisible: false,
        form: {
          name: '',
          region: '',
          date1: '',
          date2: '',
          delivery: false,
          type: [],
          resource: '',
          desc: ''
        },
        formLabelWidth: '120px'
      };
    }
  };
</script>
```

:::

Dialog imbriqué

Si un Dialog est imbriqué dans un autre Dialog, `append-to-body` est requis.

:::demo Normalement l'utilisation de Dialog imbriqué est déconseillée. Si vous avez besoin de plusieurs Dialogs sur la page, vous pouvez les aplatir afin qu'ils soit au même niveau. Si vous devez absolument utiliser un Dialog imbriqué, configurez l'attribut `append-to-body` du Dialog imbriqué à `true` et il sera ajouté au body au lieu de son noeud parent, afin d'avoir un affichage correct.

```
<template>
  <el-button type="text" @click="outerVisible = true">Ouvrir le modal extérieur</el-button>

  <el-dialog title="Modal extérieur" :visible.sync="outerVisible">
    <el-dialog
        width="30%"
        title="Modal intérieur"
        :visible.sync="innerVisible"
        append-to-body>
    </el-dialog>
    <div slot="footer" class="dialog-footer">
      <el-button @click="outerVisible = false">Annuler</el-button>
```

```
      <el-button type="primary" @click="innerVisible = true">Ouvrir le modal
intérieur</el-button>
    </div>
  </el-dialog>
</template>

<script>
  export default {
    data() {
      return {
        outerVisible: false,
        innerVisible: false
      };
    }
  }
</script>
```

:::

Centrer le contenu

Le contenu du modal peut être centré.

:::demo Régler `center` à `true` centrera horizontalement le header et le footer. `center` n'affecte que le header et le footer. Le contenu du body pouvant être n'importe quoi, si vous désirez le centrer vous devrez ajouter des règles CSS.

```
<el-button type="text" @click="centerDialogVisible = true">Cliquez pour ouvrir le
modal</el-button>

<el-dialog
  title="Attention"
  :visible.sync="centerDialogVisible"
  width="30%"
  center>
  <span>Le contenu du modal n'est pas centré par défaut.</span>
  <span slot="footer" class="dialog-footer">
    <el-button @click="centerDialogVisible = false">Annuler</el-button>
    <el-button type="primary" @click="centerDialogVisible = false">Confirmer</el-
button>
  </span>
</el-dialog>

<script>
  export default {
    data() {
      return {
        centerDialogVisible: false
      };
    }
  };
</script>
```

:::

:::tip Le contenu de Dialog bénéficie du lazy loading, ce qui signifie que le slot par défaut n'est pas généré par le DOM avant la première ouverture. Si vous avez besoin de manipuler le DOM ou d'accéder à un composant via `ref`, vous pouvez le faire avec la callback de l'évènement `open`. :::

:::tip Si la variable liée à `visible` est gérée dans Vuex, le modificateur `.sync` ne peut pas fonctionner. Dans ce cas retirez-le, écoutez les évènements `open` et `close`, et commitez les mutations Vuex pour mettre à jour la valeur de cette variable. :::

Attributs

Attribut	Description	Type	Valeurs acceptées	Défaut
visible	Visibilité du Dialog, supporte le modificateur .sync.	boolean	—	false
title	Titre du Dialog. Peut aussi être passé via un slot (voir la table suivante).	string	—	—
width	Largeur du Dialog.	string	—	50%
fullscreen	Si le Dialog doit être en plein écran.	boolean	—	false
top	Valeur du `margin-top` du CSS du Dialog.	string	—	15vh
modal	Si un masque est affiché.	boolean	—	true
modal-append-to-body	S'il faut ajouter le modal au body. Si `false`, le modal sera ajouter à l'élément parent du Dialog.	boolean	—	true
append-to-body	S'il faut ajouter le Dialog au body. Un Dialog imbriqué doit avoir cet attribut à `true`.	boolean	—	false
lock-scroll	Si le défilement du body est désactivé.	boolean	—	true
custom-class	Nom de classe pour le Dialog	string	—	—
close-on-click-modal	Si le Dialog peut être fermé en cliquant sur le masque.	boolean	—	true
close-on-press-escape	Si le Dialog peut être fermé en appuyant sur Echap.	boolean	—	true
show-close	Si le bouton de fermeture doit apparaître.	boolean	—	true
before-	Callback avant la fermeture du Dialog.	function(done)，done	—	—

close		est utilisé pour fermer le Dialog.		
center	Si le header et le footer doivent être centrés.	boolean	—	false
destroy-on-close	Destroy elements in Dialog when closed	boolean	—	false

Slot

Nom	Description
—	Contenu du Dialog.
title	Contenu du titre.
footer	Contenu du footer.

Évènements

Nom	Description	Paramètres
open	Se déclenche quand le Dialog s'ouvre.	—
opened	Se déclenche quand l'animation d'ouverture est terminée.	—
close	Se déclenche quand le Dialog se ferme.	—
closed	Se déclenche quand l'animation de fermeture du Dialog est terminée.	—