

# 目 录

致谢

## 1. 开发指南

- 1.1 安装
- 1.2 快速上手
- 1.3 国际化
- 1.4 自定义主题
- 1.5 内置过渡动画

## 2. 基本组件

- 2.2 Container布局容器
- 2.3 Color色彩
- 2.4 Typography字体
- 2.5 Icon图标
- 2.6 Button 按钮
- 2.1 Layout布局

## 3. 表单组件

- 3.1 Radio 单选框
- 3.2 Checkbox 多选框
- 3.3 Input 输入框
- 3.4 InputNumber 计数器
- 3.5 Select 选择器
- 3.6 Cascader 级联选择器
- 3.7 Switch 开关
- 3.8 Slider 滑块
- 3.9 TimePicker 时间选择器

# 致谢

当前文档《Element-UI使用手册》由 进击的皇虫 使用 书栈(BookStack.CN) 进行构建，生成于 2018-01-20。

书栈(BookStack.CN) 仅提供文档编写、整理、归类等功能，以及对文档内容的生成和导出工具。

文档内容由网友们编写和整理，书栈(BookStack.CN) 难以确认文档内容知识点是否错漏。如果您在阅读文档获取知识的时候，发现文档内容有不恰当的地方，请向我们反馈，让我们共同携手，将知识准确、高效且有效地传递给每一个人。

同时，如果您在日常生活、工作和学习中遇到有价值有营养的知识文档，欢迎分享到 书栈(BookStack.CN) ，为知识的传承献上您的一份力量！

如果当前文档生成时间太久，请到 书栈(BookStack.CN) 获取最新的文档，以跟上知识更新换代的步伐。

文档地址：<http://www.bookstack.cn/books/element-ui>

书栈官网：<http://www.bookstack.cn>

书栈开源：<https://github.com/TruthHun>

分享，让知识传承更久远！感谢知识的创造者，感谢知识的分享者，也感谢每一位阅读到此处的读者，因为我们都将成为知识的传承者。

# 1. 开发指南

## 开发指南

---

- [安装](#)
- [快速上手](#)
- [国际化](#)
- [自定义主题](#)
- [内置过渡动画](#)

## 1.1 安装

- 安装
  - npm 安装
  - CDN
  - Hello world

## 安装

### npm 安装

推荐使用 npm 的方式安装，它能更好地和 [webpack](#) 打包工具配合使用。

```
1. npm i element-ui -S
```

### CDN

目前可以通过 [unpkg.com/element-ui](https://unpkg.com/element-ui) 获取到最新版本的资源，在页面上引入 js 和 css 文件即可开始使用。

```
1. <!-- 引入样式 -->
2. <link rel="stylesheet" href="https://unpkg.com/element-
  ui/lib/theme-chalk/index.css">
3. <!-- 引入组件库 -->
4. <script src="https://unpkg.com/element-ui/lib/index.js"></script>
```

我们建议使用 CDN 引入 *Element* 的用户在链接地址上锁定版本，以免将来 *Element* 升级时受到非兼容性更新的影响。锁定版本的方法请查看 [unpkg.com](https://unpkg.com)。

## Hello world

通过 CDN 的方式我们可以很容易地使用 Element 写出一个 Hello world 页面。[在线演示](#)

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4.   <meta charset="UTF-8">
5.   <!-- 引入样式 -->
6.   <link rel="stylesheet" href="https://unpkg.com/element-
   ui/lib/theme-chalk/index.css">
7. </head>
8. <body>
9.   <div id="app">
10.    <el-button @click="visible = true">按钮</el-button>
11.    <el-dialog :visible.sync="visible" title="Hello world">
12.      <p>欢迎使用 Element</p>
13.    </el-dialog>
14.  </div>
15. </body>
16. <!-- 先引入 Vue -->
17. <script src="https://unpkg.com/vue/dist/vue.js"></script>
18. <!-- 引入组件库 -->
19. <script src="https://unpkg.com/element-ui/lib/index.js"></script>
20. <script>
21.   new Vue({
22.     el: '#app',
23.     data: function() {
24.       return { visible: false }
25.     }
26.   })
27. </script>
28. </html>
```

如果是通过 npm 安装，并希望配合 webpack 使用，请阅读下一节：快速上手。



## 1.2 快速上手

- 快速上手
  - 使用 Starter Kit
  - 使用 vue-cli
  - 引入 Element
    - 完整引入
    - 按需引入
  - 全局配置
  - 开始使用

### 快速上手

---

本节将介绍如何在项目中使用 Element。

### 使用 Starter Kit

---

我们提供了通用的[项目模板](#)，你可以直接使用。对于 Laravel 用户，我们也准备了相应的[模板](#)，同样可以直接下载使用。

如果不希望使用我们提供的模板，请继续阅读。

### 使用 vue-cli

---

我们还可以使用 [vue-cli](#) 初始化项目，命令如下：

```
1. > npm i -g vue-cli
2. > mkdir my-project && cd my-project
3. > vue init webpack
4. > npm i && npm i element-ui
```

## 引入 Element

你可以引入整个 Element，或是根据需要仅引入部分组件。我们先介绍如何引入完整的 Element。

### 完整引入

在 main.js 中写入以下内容：

```
1. import Vue from 'vue'
2. import ElementUI from 'element-ui'
3. import 'element-ui/lib/theme-chalk/index.css'
4. import App from './App.vue'
5.
6. Vue.use(ElementUI)
7.
8. new Vue({
9.   el: '#app',
10.  render: h => h(App)
11. })
```

以上代码便完成了 Element 的引入。需要注意的是，样式文件需要单独引入。

### 按需引入

借助 [babel-plugin-component](#)，我们可以只引入需要的组件，以达到减小项目体积的目的。

首先，安装 babel-plugin-component：

```
1. npm install babel-plugin-component -D
```

然后，将 .babelrc 修改为：



```

1.  {
2.    "presets": [
3.      ["es2015", { "modules": false }]
4.    ],
5.    "plugins": [["component", [
6.      {
7.        "libraryName": "element-ui",
8.        "styleLibraryName": "theme-chalk"
9.      }
10.    ]]]
11.  }

```

接下来，如果你只希望引入部分组件，比如 Button 和 Select，那么需要在 main.js 中写入以下内容：

```

1.  import Vue from 'vue'
2.  import { Button, Select } from 'element-ui'
3.  import App from './App.vue'
4.
5.  Vue.component(Button.name, Button)
6.  Vue.component(Select.name, Select)
7.  /* 或写为
8.   * Vue.use(Button)
9.   * Vue.use(Select)
10.  */
11.
12.  new Vue({
13.    el: '#app',
14.    render: h => h(App)
15.  })

```

完整组件列表和引入方式（完整组件列表以 [components.json](#) 为准）

```

1.  import Vue from 'vue'
2.  import {

```

```
3.   Pagination,
4.   Dialog,
5.   Autocomplete,
6.   Dropdown,
7.   DropdownMenu,
8.   DropdownItem,
9.   Menu,
10.  Submenu,
11.  MenuItem,
12.  MenuItemGroup,
13.  Input,
14.  InputNumber,
15.  Radio,
16.  RadioGroup,
17.  RadioButton,
18.  Checkbox,
19.  CheckboxButton,
20.  CheckboxGroup,
21.  Switch,
22.  Select,
23.  Option,
24.  OptionGroup,
25.  Button,
26.  ButtonGroup,
27.  Table,
28.  TableColumn,
29.  DatePicker,
30.  TimeSelect,
31.  TimePicker,
32.  Popover,
33.  Tooltip,
34.  Breadcrumb,
35.  BreadcrumbItem,
36.  Form,
37.  FormItem,
38.  Tabs,
39.  TabPane,
40.  Tag,
```

```
41.   Tree,  
42.   Alert,  
43.   Slider,  
44.   Icon,  
45.   Row,  
46.   Col,  
47.   Upload,  
48.   Progress,  
49.   Badge,  
50.   Card,  
51.   Rate,  
52.   Steps,  
53.   Step,  
54.   Carousel,  
55.   CarouselItem,  
56.   Collapse,  
57.   CollapseItem,  
58.   Cascader,  
59.   ColorPicker,  
60.   Transfer,  
61.   Container,  
62.   Header,  
63.   Aside,  
64.   Main,  
65.   Footer,  
66.   Loading,  
67.   MessageBox,  
68.   Message,  
69.   Notification  
70. } from 'element-ui'  
71.  
72. Vue.use(Pagination)  
73. Vue.use(Dialog)  
74. Vue.use(Autocomplete)  
75. Vue.use(Dropdown)  
76. Vue.use(DropdownMenu)  
77. Vue.use(DropdownItem)  
78. Vue.use(Menu)
```

```
79. Vue.use(Submenu)
80. Vue.use(MenuItem)
81. Vue.use(MenuItemGroup)
82. Vue.use(Input)
83. Vue.use(InputNumber)
84. Vue.use(Radio)
85. Vue.use(RadioGroup)
86. Vue.use(RadioButton)
87. Vue.use(Checkbox)
88. Vue.use(CheckboxButton)
89. Vue.use(CheckboxGroup)
90. Vue.use(Switch)
91. Vue.use(Select)
92. Vue.use(Option)
93. Vue.use(OptionGroup)
94. Vue.use(Button)
95. Vue.use(ButtonGroup)
96. Vue.use(Table)
97. Vue.use(TableColumn)
98. Vue.use(DatePicker)
99. Vue.use(TimeSelect)
100. Vue.use(TimePicker)
101. Vue.use(Popover)
102. Vue.use(Tooltip)
103. Vue.use(Breadcrumb)
104. Vue.use(BreadcrumbItem)
105. Vue.use(Form)
106. Vue.use(FormItem)
107. Vue.use(Tabs)
108. Vue.use(TabPane)
109. Vue.use(Tag)
110. Vue.use(Tree)
111. Vue.use(Alert)
112. Vue.use(Slider)
113. Vue.use(Icon)
114. Vue.use(Row)
115. Vue.use(Col)
116. Vue.use(Upload)
```

```
117. Vue.use(Progress)
118. Vue.use(Badge)
119. Vue.use(Card)
120. Vue.use(Rate)
121. Vue.use(Steps)
122. Vue.use(Step)
123. Vue.use(Carousel)
124. Vue.use(CarouselItem)
125. Vue.use(Collapse)
126. Vue.use(CollapseItem)
127. Vue.use(Cascader)
128. Vue.use(ColorPicker)
129. Vue.use(Container)
130. Vue.use(Header)
131. Vue.use(Aside)
132. Vue.use(Main)
133. Vue.use(Footer)
134.
135. Vue.use>Loading.directive)
136.
137. Vue.prototype.$loading = Loading.service
138. Vue.prototype.$msgbox = MessageBox
139. Vue.prototype.$alert = MessageBox.alert
140. Vue.prototype.$confirm = MessageBox.confirm
141. Vue.prototype.$prompt = MessageBox.prompt
142. Vue.prototype.$notify = Notification
143. Vue.prototype.$message = Message
```

## 全局配置

在引入 Element 时，可以传入一个全局配置对象。该对象目前仅支持 size 字段，用于改变组件的默认尺寸。按照引入 Element 的方式，具体操作如下：

完整引入 Element：

```
1. import Vue from 'vue'
2. import Element from 'element-ui'
3. Vue.use(Element, { size: 'small' })
```

按需引入 Element:

```
1. import Vue from 'vue'
2. import { Button } from 'element-ui'
3.
4. Vue.prototype.$ELEMENT = { size: 'small' }
5. Vue.use(Button)
```

按照以上设置，项目中所有拥有 `size` 属性的组件的默认尺寸均为 `'small'`。

## 开始使用

至此，一个基于 Vue 和 Element 的开发环境已经搭建完毕，现在就可以编写代码了。启动开发模式：

```
1. # 执行如下命令后访问 localhost:8086
2. npm run dev
```

编译：

```
1. npm run build
```

各个组件的使用方法请参阅它们各自的文档。

## 1.3 国际化

- 国际化
  - 兼容 `vue-i18n@5.x`
  - 兼容其他 `i18n` 插件
  - 兼容 `vue-i18n@6.x`
  - 按需加载里定制 `i18n`
  - 通过 `CDN` 的方式加载语言文件

## 国际化

Element 组件内部默认使用中文，若希望使用其他语言，则需要进行多语言设置。以英文为例，在 `main.js` 中：

```
1. // 完整引入 Element
2. import Vue from 'vue'
3. import ElementUI from 'element-ui'
4. import locale from 'element-ui/lib/locale/lang/en'
5.
6. Vue.use(ElementUI, { locale })
```

或

```
1. // 按需引入 Element
2. import Vue from 'vue'
3. import { Button, Select } from 'element-ui'
4. import lang from 'element-ui/lib/locale/lang/en'
5. import locale from 'element-ui/lib/locale'
6.
7. // 设置语言
8. locale.use(lang)
9.
10. // 引入组件
11. Vue.component(Button.name, Button)
```

```
12. Vue.component(Select.name, Select)
```

如果使用其它语言，默认情况下中文语言包依旧是被引入的，可以使用 webpack 的 NormalModuleReplacementPlugin 替换默认语言包。

webpack.config.js

```
1. {
2.   plugins: [
3.     new webpack.NormalModuleReplacementPlugin(/element-
4.       ui[\\/]lib[\\/]locale[\\/]lang[\\/]zh-CN/, 'element-
5.       ui/lib/locale/lang/en')
6.   ]
7. }
```

## 兼容 vue-i18n@5.x

Element 兼容 vue-i18n@5.x，搭配使用能更方便地实现多语言切换。

```
1. import Vue from 'vue'
2. import VueI18n from 'vue-i18n'
3. import Element from 'element-ui'
4. import enLocale from 'element-ui/lib/locale/lang/en'
5. import zhLocale from 'element-ui/lib/locale/lang/zh-CN'
6.
7. Vue.use(VueI18n)
8. Vue.use(Element)
9.
10. Vue.config.lang = 'zh-cn'
11. Vue.locale('zh-cn', zhLocale)
12. Vue.locale('en', enLocale)
```



## 兼容其他 i18n 插件

如果不使用 `vue-i18n@5.x`，而是用其他的 i18n 插件，Element 将无法兼容，但是可以自定义 Element 的 i18n 的处理方法。

```
1. import Vue from 'vue'
2. import Element from 'element-ui'
3. import enLocale from 'element-ui/lib/locale/lang/en'
4. import zhLocale from 'element-ui/lib/locale/lang/zh-CN'
5.
6. Vue.use(Element, {
7.   i18n: function (path, options) {
8.     // ...
9.   }
10. })
```

## 兼容 vue-i18n@6.x

默认不支持 6.x 的 vue-i18n，你需要手动处理。

```
1. import Vue from 'vue'
2. import Element from 'element-ui'
3. import VueI18n from 'vue-i18n'
4. import enLocale from 'element-ui/lib/locale/lang/en'
5. import zhLocale from 'element-ui/lib/locale/lang/zh-CN'
6.
7. Vue.use(VueI18n)
8.
9. const messages = {
10.   en: {
11.     message: 'hello',
12.     ...enLocale // 或者用 Object.assign({ message: 'hello' },
13.       enLocale)
14.   },
15.   zh: {
16.     message: '你好',
```

```

16.     ...zhLocale // 或者用 Object.assign({ message: '你好' },
    zhLocale)
17.   }
18. }
19. // Create VueI18n instance with options
20. const i18n = new VueI18n({
21.   locale: 'en', // set locale
22.   messages, // set locale messages
23. })
24.
25. Vue.use(Element, {
26.   i18n: (key, value) => i18n.t(key, value)
27. })
28.
29. new Vue({ i18n }).$mount('#app')

```

## 按需加载里定制 i18n

```

1. import Vue from 'vue'
2. import DatePicker from 'element/lib/date-picker'
3. import VueI18n from 'vue-i18n'
4.
5. import enLocale from 'element-ui/lib/locale/lang/en'
6. import zhLocale from 'element-ui/lib/locale/lang/zh-CN'
7. import ElementLocale from 'element-ui/lib/locale'
8.
9. Vue.use(VueI18n)
10. Vue.use(DatePicker)
11.
12. const messages = {
13.   en: {
14.     message: 'hello',
15.     ...enLocale
16.   },
17.   zh: {
18.     message: '你好',
19.     ...zhLocale

```

```

20.   }
21. }
22. // Create VueI18n instance with options
23. const i18n = new VueI18n({
24.   locale: 'en', // set locale
25.   messages, // set locale messages
26. })
27.
28. ElementLocale.i18n((key, value) => i18n.t(key, value))

```

## 通过 CDN 的方式加载语言文件

```

1. <script src="//unpkg.com/vue"></script>
2. <script src="//unpkg.com/element-ui"></script>
3. <script src="//unpkg.com/element-ui/lib/umd/locale/en.js"></script>
4.
5. <script>
6.   ELEMENT.locale(ELEMENT.lang.en)
7. </script>

```

### 搭配 vue-i18n 使用

```

1. <script src="//unpkg.com/vue"></script>
2. <script src="//unpkg.com/vue-i18n/dist/vue-i18n.js"></script>
3. <script src="//unpkg.com/element-ui"></script>
4. <script src="//unpkg.com/element-ui/lib/umd/locale/zh-CN.js">
   </script>
5. <script src="//unpkg.com/element-ui/lib/umd/locale/en.js"></script>
6.
7. <script>
8.   Vue.locale('en', ELEMENT.lang.en)
9.   Vue.locale('zh-cn', ELEMENT.lang.zhCN)
10. </script>

```

目前 Element 内置了以下语言：

- 简体中文 ( zh-CN )
- 英语 ( en )
- 德语 ( de )
- 葡萄牙语 ( pt )
- 西班牙语 ( es )
- 丹麦语 ( da )
- 法语 ( fr )
- 挪威语 ( nb-NO )
- 繁体中文 ( zh-TW )
- 意大利语 ( it )
- 韩语 ( ko )
- 日语 ( ja )
- 荷兰语 ( nl )
- 越南语 ( vi )
- 俄语 ( ru-RU )
- 土耳其语 ( tr-TR )
- 巴西葡萄牙语 ( pt-br )
- 波斯语 ( fa )
- 泰语 ( th )
- 印尼语 ( id )
- 保加利亚语 ( bg )
- 波兰语 ( pl )
- 芬兰语 ( fi )
- 瑞典语 ( sv-SE )
- 希腊语 ( el )
- 斯洛伐克语 ( sk )
- 加泰罗尼亚语 ( ca )
- 捷克语 ( cz )

- 乌克兰语 ( ua )
- 土库曼语 ( tk )
- 泰米尔语 ( ta )
- 拉脱维亚语 ( lv )
- 南非荷兰语 ( af-ZA )
- 爱沙尼亚语 ( ee )
- 斯洛文尼亚语 ( sl )
- 阿拉伯语 ( ar )
- 希伯来语 ( he )
- 立陶宛语 ( lt )
- 蒙古语 ( mn )
- 哈萨克斯坦语 ( kz )
- 匈牙利语 ( hu )
- 罗马尼亚语 ( ro )

如果你需要使用其他的语言，欢迎贡献 PR：只需在 [这里](#) 添加一个语言配置文件即可。

## 1.4 自定义主题

- 自定义主题
  - 仅替换主题色
  - 在项目中改变 SCSS 变量
  - 命令行主题工具
  - 安装工具
  - 初始化变量文件
  - 修改变量
  - 编译主题
  - 引入自定义主题
  - 搭配插件按需引入组件主题

### 自定义主题

---

Element 默认提供一套主题，CSS 命名采用 BEM 的风格，方便使用者覆盖样式。我们提供了三种方法，可以进行不同程度的样式自定义。

#### 仅替换主题色

---

如果仅希望更换 Element 的主题色，推荐使用在线主题生成工具。Element 默认的主题色是鲜艳、友好的蓝色。通过替换主题色，能够让 Element 的视觉更加符合具体项目的定位。

使用上述工具，可以很方便地实时预览主题色改变之后的视觉，同时它还可以基于新的主题色生成完整的样式文件包，供直接下载使用（关于如何使用下载的主题包，请参考本节「引入自定义主题」和「搭配插件按需引入组件主题」部分）。

#### 在项目中改变 SCSS 变量

Element 的 theme-chalk 使用 SCSS 编写，如果你的项目也使用了 SCSS，那么可以直接在项目中改变 Element 的样式变量。新建一个样式文件，例如 element-variables.scss，写入以下内容：

```
1.  /* 改变主题色变量 */
2.  $--color-primary: teal;
3.
4.  /* 改变 icon 字体路径变量，必需 */
5.  $--font-path: '~element-ui/lib/theme-chalk/fonts';
6.
7.  @import "~element-ui/packages/theme-chalk/src/index";
```

之后，在项目的入口文件中，直接引入以上样式文件即可（无需引入 Element 编译好的 CSS 文件）：

```
1.  import Vue from 'vue'
2.  import Element from 'element-ui'
3.  import './element-variables.scss'
4.
5.  Vue.use(Element)
```

需要注意的是，覆盖字体路径变量是必需的，将其赋值为 Element 中 icon 图标所在的相对路径即可。需要注意的是，覆盖字体路径变量是必需的，将其赋值为 Element 中 icon 图标所在的相对路径即可。

## 命令行主题工具

如果你的项目没有使用 SCSS，那么可以使用命令行主题工具进行深层次的主题定制：

## 安装工具

首先安装「主题生成工具」，可以全局安装或者安装在当前项目下，推

荐安装在项目里，方便别人 clone 项目时能直接安装依赖并启动，这里以全局安装做演示。

```
1. npm i element-theme -g
```

安装白垩主题，可以从 npm 安装或者从 GitHub 拉取最新代码。

```
1. # 从 npm
2. npm i element-theme-chalk -D
3.
4. # 从 GitHub
5. npm i https://github.com/ElementUI/theme-chalk -D
```

## 初始化变量文件

主题生成工具安装成功后，如果全局安装可以在命令行里通过 `et` 调用工具，如果安装在当前目录下，需要通过 `node_modules/.bin/et` 访问到命令。执行 `-i` 初始化变量文件。默认输出到 `element-variables.scss`，当然你可以传参数指定文件输出目录。

```
1. et -i [可以自定义变量文件]
2.
3. > ✓ Generator variables file
```

如果使用默认配置，执行后当前目录会有一个 `element-variables.scss` 文件。内部包含了主题所用到的所有变量，它们使用 SCSS 的格式定义。大致结构如下：

```
1. $--color-primary: #409EFF !default;
2. $--color-primary-light-1: mix($--color-white, $--color-primary, 10%) !default; /* 53a8ff */
3. $--color-primary-light-2: mix($--color-white, $--color-primary, 20%) !default; /* 66b1ff */
```



```

4. $--color-primary-light-3: mix($--color-white, $--color-primary,
  30%) !default; /* 79bbff */
5. $--color-primary-light-4: mix($--color-white, $--color-primary,
  40%) !default; /* 8cc5ff */
6. $--color-primary-light-5: mix($--color-white, $--color-primary,
  50%) !default; /* a0cfff */
7. $--color-primary-light-6: mix($--color-white, $--color-primary,
  60%) !default; /* b3d8ff */
8. $--color-primary-light-7: mix($--color-white, $--color-primary,
  70%) !default; /* c6e2ff */
9. $--color-primary-light-8: mix($--color-white, $--color-primary,
  80%) !default; /* d9ecff */
10. $--color-primary-light-9: mix($--color-white, $--color-primary,
  90%) !default; /* ecf5ff */
11.
12. $--color-success: #67c23a !default;
13. $--color-warning: #e6a23c !default;
14. $--color-danger: #f56c6c !default;
15. $--color-info: #909399 !default;
16.
17. ...

```

## 修改变量

直接编辑 `element-variables.scss` 文件，例如修改主题色为红色。

```
1. $--color-primary: red;
```

## 编译主题

保存文件后，到命令行里执行 `et` 编译主题，如果你想启用 `watch` 模式，实时编译主题，增加 `-w` 参数；如果你在初始化时指定了自定义变量文件，则需要增加 `-c` 参数，并带上你的变量文件名

1. `et`
- 2.
3. `> ✓ build theme font`
4. `> ✓ build element theme`

## 引入自定义主题

默认情况下编译的主题目录是放在 `./theme` 下，你可以通过 `-o` 参数指定打包目录。像引入默认主题一样，在代码里直接引用 `theme/index.css` 文件即可。

```
1. import '../theme/index.css'
2. import ElementUI from 'element-ui'
3. import Vue from 'vue'
4.
5. Vue.use(ElementUI)
```

## 搭配插件按需引入组件主题

如果是搭配 `babel-plugin-component` 一起使用，只需要修改 `.babelrc` 的配置，指定 `styleLibraryName` 路径为自定义主题相对于 `.babelrc` 的路径，注意要加 `~`。

```
1. {
2.   "plugins": [["component", [
3.     {
4.       "libraryName": "element-ui",
5.       "styleLibraryName": "~theme"
6.     }
7.   ]]]
8. }
```

如果不清楚 `babel-plugin-component` 是什么，请阅读 [快速上手](#) 一

节。更多 `element-theme` 用法请参考项目仓库。

## 1.5 内置过渡动画

- [内置过渡动画](#)
  - [fade](#) 淡入淡出
  - [collapse](#) 展开折叠
  - [按需引入](#)

### 内置过渡动画

Element 内应用在部分组件的过渡动画，你也可以直接使用。在使用之前请阅读 [transition 组件文档](#)。

### fade 淡入淡出

提供 `el-fade-in-linear` 和 `el-fade-in` 两种效果。

```

1. <template>
2.   <div>
3.     <el-button @click="show = !show">Click Me</el-button>
4.
5.     <div style="display: flex; margin-top: 20px; height: 100px;">
6.       <transition name="el-fade-in-linear">
7.         <div v-show="show" class="transition-box">.el-fade-in-
linear</div>
8.       </transition>
9.       <transition name="el-fade-in">
10.        <div v-show="show" class="transition-box">.el-fade-in</div>
11.      </transition>
12.    </div>
13.  </div>
14. </template>
15.
16. <script>
17.   export default {

```

```

18.     data: () => ({
19.         show: true
20.     })
21. }
22. </script>
23.
24. <style>
25.     .transition-box {
26.         margin-bottom: 10px;
27.         width: 200px;
28.         height: 100px;
29.         border-radius: 4px;
30.         background-color: #409EFF;
31.         text-align: center;
32.         color: #fff;
33.         padding: 40px 20px;
34.         box-sizing: border-box;
35.         margin-right: 20px;
36.     }
37. </style>
38. ¶ zoom 缩放
39. .el-zoom-in-center
40. .el-zoom-in-top
41. .el-zoom-in-bottom
42. 提供 el-zoom-in-center, el-zoom-in-top 和 el-zoom-in-bottom 三种效果。
43.
44. <template>
45.     <div>
46.         <el-button @click="show2 = !show2">Click Me</el-button>
47.
48.         <div style="display: flex; margin-top: 20px; height: 100px;">
49.             <transition name="el-zoom-in-center">
50.                 <div v-show="show2" class="transition-box">.el-zoom-in-
center</div>
51.             </transition>
52.
53.             <transition name="el-zoom-in-top">
54.                 <div v-show="show2" class="transition-box">.el-zoom-in-

```

```

    top</div>
55.     </transition>
56.
57.     <transition name="el-zoom-in-bottom">
58.         <div v-show="show2" class="transition-box">.el-zoom-in-
    bottom</div>
59.     </transition>
60. </div>
61. </div>
62. </template>
63.
64. <script>
65.     export default {
66.         data: () => ({
67.             show2: true
68.         })
69.     }
70. </script>
71.
72. <style>
73.     .transition-box {
74.         margin-bottom: 10px;
75.         width: 200px;
76.         height: 100px;
77.         border-radius: 4px;
78.         background-color: #409EFF;
79.         text-align: center;
80.         color: #fff;
81.         padding: 40px 20px;
82.         box-sizing: border-box;
83.         margin-right: 20px;
84.     }
85. </style>

```

## collapse 展开折叠

使用 `el-collapse-transition` 组件实现折叠展开效果。

```
1. <template>
2.   <div>
3.     <el-button @click="show3 = !show3">Click Me</el-button>
4.
5.     <div style="margin-top: 20px; height: 200px;">
6.       <el-collapse-transition>
7.         <div v-show="show3">
8.           <div class="transition-box">el-collapse-transition</div>
9.           <div class="transition-box">el-collapse-transition</div>
10.        </div>
11.      </el-collapse-transition>
12.    </div>
13.  </div>
14. </template>
15.
16. <script>
17.   export default {
18.     data: () => ({
19.       show3: true
20.     })
21.   }
22. </script>
23.
24. <style>
25.   .transition-box {
26.     margin-bottom: 10px;
27.     width: 200px;
28.     height: 100px;
29.     border-radius: 4px;
30.     background-color: #409EFF;
31.     text-align: center;
32.     color: #fff;
33.     padding: 40px 20px;
34.     box-sizing: border-box;
35.     margin-right: 20px;
36.   }
37. </style>
```

## 按需引入

---

```
1. // fade/zoom 等
2. import 'element-ui/lib/theme-chalk/base.css';
3. // collapse 展开折叠
4. import CollapseTransition from 'element-
   ui/lib/transitions/collapse-transition';
5. import Vue from 'vue'
6.
7. Vue.component(CollapseTransition.name, CollapseTransition)
```



## 2. 基本组件

## 2.2 Container布局容器

- Container 布局容器
  - 常见页面布局
  - 实例
  - Container Attributes
  - Header Attributes
  - Aside Attributes
  - Footer Attributes

### Container 布局容器

用于布局的容器组件，方便快速搭建页面的基本结构：

`<el-container>`：外层容器。当子元素中包含 `<el-header>` 或 `<el-footer>` 时，全部子元素会垂直上下排列，否则会水平左右排列。

`<el-header>`：顶栏容器。

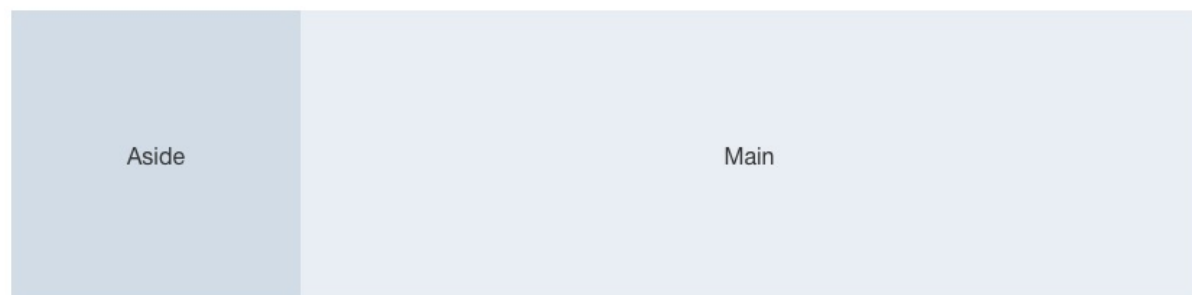
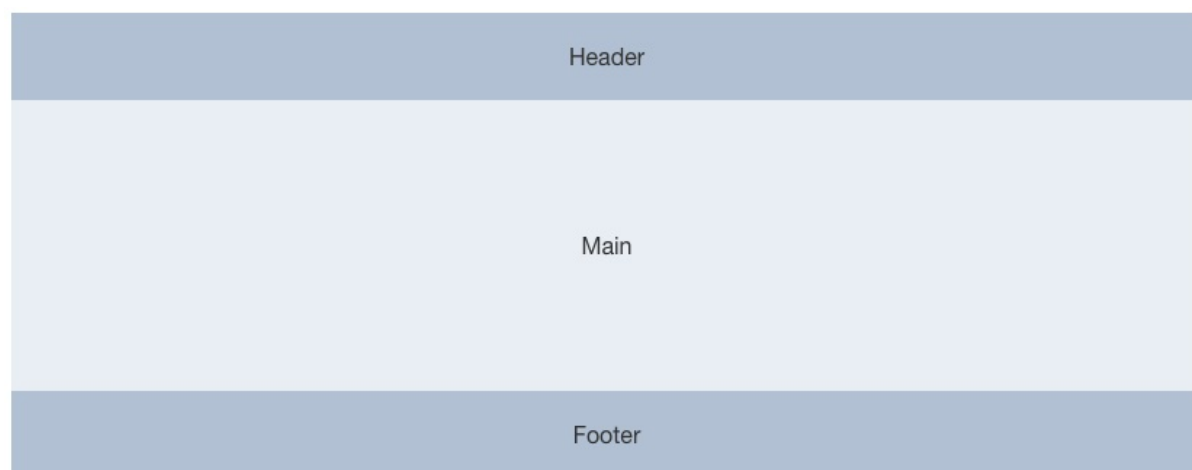
`<el-aside>`：侧边栏容器。

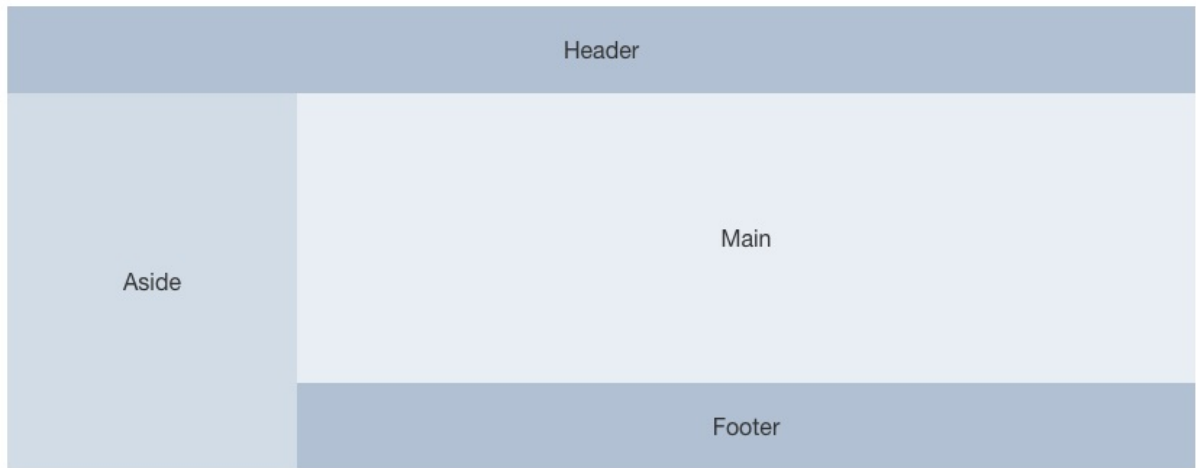
`<el-main>`：主要区域容器。

`<el-footer>`：底栏容器。

以上组件采用了 `flex` 布局，使用前请确定目标浏览器是否兼容。此外，`<el-container>` 的子元素只能是后四者，后四者的父元素也只能是 `<el-container>`。

### 常见页面布局





```
1. <el-container>
2.   <el-header>Header</el-header>
3.   <el-main>Main</el-main>
4. </el-container>
5.
6. <el-container>
7.   <el-header>Header</el-header>
```

```
8.   <el-main>Main</el-main>
9.   <el-footer>Footer</el-footer>
10. </el-container>
11.
12. <el-container>
13.   <el-aside width="200px">Aside</el-aside>
14.   <el-main>Main</el-main>
15. </el-container>
16.
17. <el-container>
18.   <el-header>Header</el-header>
19.   <el-container>
20.     <el-aside width="200px">Aside</el-aside>
21.     <el-main>Main</el-main>
22.   </el-container>
23. </el-container>
24.
25. <el-container>
26.   <el-header>Header</el-header>
27.   <el-container>
28.     <el-aside width="200px">Aside</el-aside>
29.     <el-container>
30.       <el-main>Main</el-main>
31.       <el-footer>Footer</el-footer>
32.     </el-container>
33.   </el-container>
34. </el-container>
35.
36. <el-container>
37.   <el-aside width="200px">Aside</el-aside>
38.   <el-container>
39.     <el-header>Header</el-header>
40.     <el-main>Main</el-main>
41.   </el-container>
42. </el-container>
43.
44. <el-container>
45.   <el-aside width="200px">Aside</el-aside>
```

```
46.   <el-container>
47.     <el-header>Header</el-header>
48.     <el-main>Main</el-main>
49.     <el-footer>Footer</el-footer>
50.   </el-container>
51. </el-container>
52.
53. <style>
54.   .el-header, .el-footer {
55.     background-color: #B3C0D1;
56.     color: #333;
57.     text-align: center;
58.     line-height: 60px;
59.   }
60.
61.   .el-aside {
62.     background-color: #D3DCE6;
63.     color: #333;
64.     text-align: center;
65.     line-height: 200px;
66.   }
67.
68.   .el-main {
69.     background-color: #E9EEF3;
70.     color: #333;
71.     text-align: center;
72.     line-height: 160px;
73.   }
74.
75.   body > .el-container {
76.     margin-bottom: 40px;
77.   }
78.
79.   .el-container:nth-child(5) .el-aside,
80.   .el-container:nth-child(6) .el-aside {
81.     line-height: 260px;
82.   }
83.
```

```
84.     .el-container:nth-child(7) .el-aside {
85.         line-height: 320px;
86.     }
87. </style>
```

## 实例

```

1. <el-container style="height: 500px; border: 1px solid #eee">
2.   <el-aside width="200px" style="background-color: rgb(238, 241,
   246)">
3.     <el-menu :default-openeds="['1', '3']">
4.       <el-submenu index="1">
5.         <template slot="title"><i class="el-icon-message"></i>导航
   一</template>
6.         <el-menu-item-group>
7.           <template slot="title">分组一</template>
8.           <el-menu-item index="1-1">选项1</el-menu-item>
9.           <el-menu-item index="1-2">选项2</el-menu-item>
10.        </el-menu-item-group>
11.        <el-menu-item-group title="分组2">
12.          <el-menu-item index="1-3">选项3</el-menu-item>
13.        </el-menu-item-group>
14.        <el-submenu index="1-4">
15.          <template slot="title">选项4</template>
16.          <el-menu-item index="1-4-1">选项4-1</el-menu-item>
17.        </el-submenu>
18.      </el-submenu>
19.      <el-submenu index="2">
20.        <template slot="title"><i class="el-icon-menu"></i>导航二
   </template>
21.        <el-menu-item-group>
22.          <template slot="title">分组一</template>
23.          <el-menu-item index="2-1">选项1</el-menu-item>
24.          <el-menu-item index="2-2">选项2</el-menu-item>
25.        </el-menu-item-group>
26.        <el-menu-item-group title="分组2">
27.          <el-menu-item index="2-3">选项3</el-menu-item>

```

```

28.         </el-menu-item-group>
29.         <el-submenu index="2-4">
30.             <template slot="title">选项4</template>
31.             <el-menu-item index="2-4-1">选项4-1</el-menu-item>
32.         </el-submenu>
33.     </el-submenu>
34.     <el-submenu index="3">
35.         <template slot="title"><i class="el-icon-setting"></i>导航
    三</template>
36.         <el-menu-item-group>
37.             <template slot="title">分组一</template>
38.             <el-menu-item index="3-1">选项1</el-menu-item>
39.             <el-menu-item index="3-2">选项2</el-menu-item>
40.         </el-menu-item-group>
41.         <el-menu-item-group title="分组2">
42.             <el-menu-item index="3-3">选项3</el-menu-item>
43.         </el-menu-item-group>
44.         <el-submenu index="3-4">
45.             <template slot="title">选项4</template>
46.             <el-menu-item index="3-4-1">选项4-1</el-menu-item>
47.         </el-submenu>
48.     </el-submenu>
49. </el-menu>
50. </el-aside>
51.
52. <el-container>
53.     <el-header style="text-align: right; font-size: 12px">
54.         <el-dropdown>
55.             <i class="el-icon-setting" style="margin-right: 15px"></i>
56.             <el-dropdown-menu slot="dropdown">
57.                 <el-dropdown-item>查看</el-dropdown-item>
58.                 <el-dropdown-item>新增</el-dropdown-item>
59.                 <el-dropdown-item>删除</el-dropdown-item>
60.             </el-dropdown-menu>
61.         </el-dropdown>
62.         <span>王小虎</span>
63.     </el-header>
64.

```



```
65.     <el-main>
66.       <el-table :data="tableData">
67.         <el-table-column prop="date" label="日期" width="140">
68.         </el-table-column>
69.         <el-table-column prop="name" label="姓名" width="120">
70.         </el-table-column>
71.         <el-table-column prop="address" label="地址">
72.         </el-table-column>
73.       </el-table>
74.     </el-main>
75.   </el-container>
76. </el-container>
77.
78. <style>
79.   .el-header {
80.     background-color: #B3C0D1;
81.     color: #333;
82.     line-height: 60px;
83.   }
84.
85.   .el-aside {
86.     color: #333;
87.   }
88. </style>
89.
90. <script>
91.   export default {
92.     data() {
93.       const item = {
94.         date: '2016-05-02',
95.         name: '王小虎',
96.         address: '上海市普陀区金沙江路 1518 弄'
97.       };
98.       return {
99.         tableData: Array(20).fill(item)
100.      }
101.    }
102.  };
```

```
103. </script>
```

## Container Attributes

参数	说明	类型	可选值	默认值
direction	子元素的排列方向	string	horizontal / vertical	子元素中有 el-header 或 el-footer 时为 vertical，否则为 horizontal

## Header Attributes

参数	说明	类型	可选值	默认值
height	顶栏高度	string	—	60px

## Aside Attributes

参数	说明	类型	可选值	默认值
width	侧边栏宽度	string	—	300px

## Footer Attributes

参数	说明	类型	可选值	默认值
height	底栏高度	string	—	60px

## 2.3 Color色彩

- Color 色彩
  - 主色
  - 辅助色
  - 中性色

### Color 色彩

---

Element 为了避免视觉传达差异，使用一套特定的调色板来规定颜色，为你所搭建的产品提供一致的外观视觉感受。

### 主色

---

Element 主要品牌颜色是鲜艳、友好的蓝色。

- Blue #409EFF

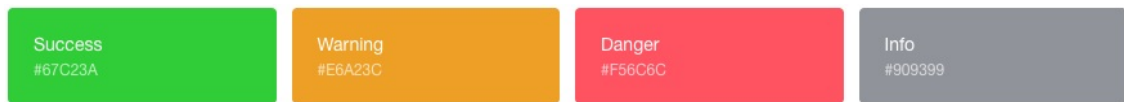


### 辅助色

---

除了主色外的场景色，需要在不同的场景中使用（例如危险色表示危险的操作）。

- Success #67C23A
- Warning #E6A23C
- Danger #F56C6C
- Info #909399



## 中性色

中性色用于文本、背景和边框颜色。通过运用不同的中性色，来表现层次结构。

- 主要文字 #303133
- 常规文字 #606266
- 次要文字 #909399
- 占位文字 #C0C4CC
- 一级边框 #DCDFE6
- 二级边框 #E4E7ED
- 三级边框 #EBEEF5
- 四级边框 #F2F6FC





## 2.4 Typography字体

- [Typography 字体](#)
  - [中文字体](#)
  - [英文 / 数字字体](#)
  - [Font-family 代码](#)
  - [字体使用规范](#)

### Typography 字体

---

我们对字体进行统一规范，力求在各个操作系统下都有最佳展示效果。

### 中文字体

---

- PingFang SC



- Hiragino Sans GB



- Microsoft YaHei



## 英文 / 数字字体

---

- Helvetica Neue



- Helvetica



- Arial



## Font-family 代码

```
1. font-family: "Helvetica Neue", Helvetica, "PingFang SC", "Hiragino Sans GB", "Microsoft YaHei", "微软雅黑", Arial, sans-serif;
```

## 字体使用规范

名称	大小
主标题	20px Extra large
标题	18px large
小标题	16px Medium
正文	14px Small
正文（小）	13px Extra Small
辅助文字	12px Extra Extra Small





## 2.5 Icon图标

- [Icon 图标](#)
  - [使用方法](#)
  - [图标集合](#)

### Icon 图标

提供了一套常用的图标集合。





































### 使用方法



































直接通过设置类名为 `el-icon-iconName` 来使用即可。例如：



1. `<i class="el-icon-edit"></i>`
2. `<i class="el-icon-share"></i>`
3. `<i class="el-icon-delete"></i>`
4. `<el-button type="primary" icon="el-icon-search">搜索</el-button>`

### 图标集合

					
el-icon-upload	el-icon-error	el-icon-success	el-icon-warning	el-icon-sort-down	el-icon-sort-up
					
el-icon-arrow-left	el-icon-circle-plus	el-icon-circle-plus-outline	el-icon-arrow-down	el-icon-arrow-right	el-icon-arrow-up
					
el-icon-back	el-icon-circle-close	el-icon-date	el-icon-circle-close-outline	el-icon-caret-left	el-icon-caret-bottom
					
el-icon-caret-top	el-icon-caret-right	el-icon-close	el-icon-d-arrow-left	el-icon-check	el-icon-delete
					
el-icon-d-arrow-right	el-icon-document	el-icon-d-caret	el-icon-edit-outline	el-icon-download	el-icon-goods
					
el-icon-search	el-icon-info	el-icon-message	el-icon-edit	el-icon-location	el-icon-loading

					
el-icon-location-outline	el-icon-menu	el-icon-minus	el-icon-bell	el-icon-mobile-phone	el-icon-news
					
el-icon-more	el-icon-more-outline	el-icon-phone	el-icon-phone-outline	el-icon-picture	el-icon-picture-outline
					
el-icon-plus	el-icon-printer	el-icon-rank	el-icon-refresh	el-icon-question	el-icon-remove
					
el-icon-share	el-icon-star-on	el-icon-setting	el-icon-circle-check	el-icon-service	el-icon-sold-out
					
el-icon-remove-outline	el-icon-star-off	el-icon-circle-check-outline	el-icon-tickets	el-icon-sort	el-icon-zoom-in
					
el-icon-time	el-icon-view	el-icon-upload2	el-icon-zoom-out		

## 2.6 Button 按钮

- Button 按钮
  - 基础用法
  - 禁用状态
  - 文字按钮
  - 图标按钮
  - 按钮组
  - 加载中
  - 不同尺寸
  - Attributes

### Button 按钮

常用的操作按钮。

### 基础用法

基础的按钮用法。



使用 `type`、`plain` 和 `round` 属性来定义 Button 的样式。

```
1. <div>
2.   <el-button>默认按钮</el-button>
3.   <el-button type="primary">主要按钮</el-button>
4.   <el-button type="success">成功按钮</el-button>
```

```

5.   <el-button type="info">信息按钮</el-button>
6.   <el-button type="warning">警告按钮</el-button>
7.   <el-button type="danger">危险按钮</el-button>
8. </div>
9.
10. <div style="margin: 20px 0">
11.   <el-button plain>朴素按钮</el-button>
12.   <el-button type="primary" plain>主要按钮</el-button>
13.   <el-button type="success" plain>成功按钮</el-button>
14.   <el-button type="info" plain>信息按钮</el-button>
15.   <el-button type="warning" plain>警告按钮</el-button>
16.   <el-button type="danger" plain>危险按钮</el-button>
17. </div>
18.
19. <div>
20.   <el-button round>圆形按钮</el-button>
21.   <el-button type="primary" round>主要按钮</el-button>
22.   <el-button type="success" round>成功按钮</el-button>
23.   <el-button type="info" round>信息按钮</el-button>
24.   <el-button type="warning" round>警告按钮</el-button>
25.   <el-button type="danger" round>危险按钮</el-button>
26. </div>

```

## 禁用状态

按钮不可用状态。



你可以使用 `disabled` 属性来定义按钮是否可用，它接受一个Boolean值。

```

1. <div>
2.   <el-button disabled>默认按钮</el-button>

```

```

3.   <el-button type="primary" disabled>主要按钮</el-button>
4.   <el-button type="success" disabled>成功按钮</el-button>
5.   <el-button type="info" disabled>信息按钮</el-button>
6.   <el-button type="warning" disabled>警告按钮</el-button>
7.   <el-button type="danger" disabled>危险按钮</el-button>
8. </div>
9.
10. <div style="margin-top: 20px">
11.   <el-button plain disabled>朴素按钮</el-button>
12.   <el-button type="primary" plain disabled>主要按钮</el-button>
13.   <el-button type="success" plain disabled>成功按钮</el-button>
14.   <el-button type="info" plain disabled>信息按钮</el-button>
15.   <el-button type="warning" plain disabled>警告按钮</el-button>
16.   <el-button type="danger" plain disabled>危险按钮</el-button>
17. </div>

```

## 文字按钮

没有边框和背景色的按钮。

```

1. <el-button type="text">文字按钮</el-button>
2. <el-button type="text" disabled>文字按钮</el-button>

```

## 图标按钮

带图标的按钮可增强辨识度（有文字）或节省空间（无文字）。



设置 `icon` 属性即可，`icon` 的列表可以参考 Element 的 `icon` 组件，也可以设置在文字右边的 `icon`，只要使用 `i` 标签即可，可以使用自定义图标。

```

1. <el-button type="primary" icon="el-icon-edit"></el-button>

```

```

2. <el-button type="primary" icon="el-icon-share"></el-button>
3. <el-button type="primary" icon="el-icon-delete"></el-button>
4. <el-button type="primary" icon="el-icon-search">搜索</el-button>
5. <el-button type="primary">上传<i class="el-icon-upload el-icon--right"></i></el-button>

```

## 按钮组

以按钮组的方式出现，常用于多项类似操作。



使用 `<el-button-group>` 标签来嵌套你的按钮。

```

1. <el-button-group>
2.   <el-button type="primary" icon="el-icon-arrow-left">上一页</el-button>
3.   <el-button type="primary">下一页<i class="el-icon-arrow-right el-icon--right"></i></el-button>
4. </el-button-group>
5. <el-button-group>
6.   <el-button type="primary" icon="el-icon-edit"></el-button>
7.   <el-button type="primary" icon="el-icon-share"></el-button>
8.   <el-button type="primary" icon="el-icon-delete"></el-button>
9. </el-button-group>

```

## 加载中

点击按钮后进行数据加载操作，在按钮上显示加载状态。



要设置为 `loading` 状态，只要设置 `loading` 属性为 `true` 即可。



```
1. <el-button type="primary" :loading="true">加载中</el-button>
```

## 不同尺寸

Button 组件提供除了默认值以外的三种尺寸，可以在不同场景下选择合适的按钮尺寸。



额外的尺寸：`medium`、`small`、`mini`，通过设置 `size` 属性来配置它们。

```
1. <div>
2.   <el-button>默认按钮</el-button>
3.   <el-button size="medium">中等按钮</el-button>
4.   <el-button size="small">小型按钮</el-button>
5.   <el-button size="mini">超小按钮</el-button>
6. </div>
7. <div style="margin-top: 20px">
8.   <el-button round>默认按钮</el-button>
9.   <el-button size="medium" round>中等按钮</el-button>
10.  <el-button size="small" round>小型按钮</el-button>
11.  <el-button size="mini" round>超小按钮</el-button>
12. </div>
```

## Attributes

参数	说明	类型	可选值	默认值
size	尺寸	string	medium / small / mini	—
type	类型	string	primary / success / warning / danger / info / text	—
	是否朴素			

	按钮			
round	是否圆形按钮	boolean	–	false
loading	是否加载中状态	boolean	–	false
disabled	是否禁用状态	boolean	–	false
icon	图标类名	string	–	–
autofocus	是否默认聚焦	boolean	–	false
native-type	原生 type 属性	string	button / submit / reset	button

## 2.1 Layout布局

- Layout 布局
  - 基础布局
  - 分栏间隔
  - 混合布局
  - 分栏偏移
  - 对齐方式
  - 响应式布局
  - 基于断点的隐藏类
  - Row Attributes
  - Col Attributes

### Layout 布局

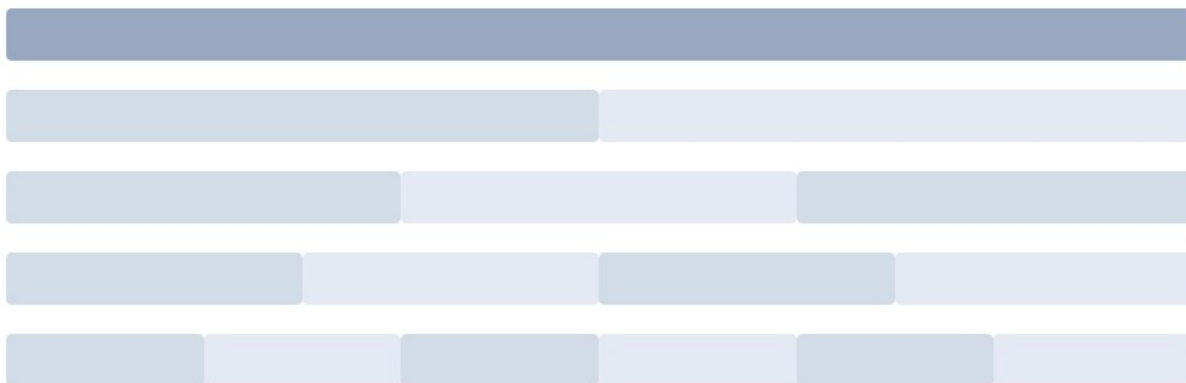
---

通过基础的 24 分栏，迅速简便地创建布局。

### 基础布局

---

使用单一分栏创建基础的栅格布局。



通过 row 和 col 组件，并通过 col 组件的 span 属性我们就可以自由地组合布局。

```
1. <el-row>
2.   <el-col :span="24"><div class="grid-content bg-purple-dark">
      </div></el-col>
3. </el-row>
4. <el-row>
5.   <el-col :span="12"><div class="grid-content bg-purple"></div>
      </el-col>
6.   <el-col :span="12"><div class="grid-content bg-purple-light">
      </div></el-col>
7. </el-row>
8. <el-row>
9.   <el-col :span="8"><div class="grid-content bg-purple"></div></el-
      col>
10.  <el-col :span="8"><div class="grid-content bg-purple-light">
      </div></el-col>
11.  <el-col :span="8"><div class="grid-content bg-purple"></div></el-
      col>
12. </el-row>
13. <el-row>
14.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
      col>
15.   <el-col :span="6"><div class="grid-content bg-purple-light">
      </div></el-col>
16.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
      col>
17.   <el-col :span="6"><div class="grid-content bg-purple-light">
      </div></el-col>
18. </el-row>
19. <el-row>
20.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
      col>
21.   <el-col :span="4"><div class="grid-content bg-purple-light">
      </div></el-col>
22.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
      col>
23.   <el-col :span="4"><div class="grid-content bg-purple-light">
      </div></el-col>
24.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
```

```
col>
25.   <el-col :span="4"><div class="grid-content bg-purple-light">
      </div></el-col>
26. </el-row>
27.
28. <style>
29.   .el-row {
30.     margin-bottom: 20px;
31.     &:last-child {
32.       margin-bottom: 0;
33.     }
34.   }
35.   .el-col {
36.     border-radius: 4px;
37.   }
38.   .bg-purple-dark {
39.     background: #99a9bf;
40.   }
41.   .bg-purple {
42.     background: #d3dce6;
43.   }
44.   .bg-purple-light {
45.     background: #e5e9f2;
46.   }
47.   .grid-content {
48.     border-radius: 4px;
49.     min-height: 36px;
50.   }
51.   .row-bg {
52.     padding: 10px 0;
53.     background-color: #f9fafc;
54.   }
55. </style>
```

## 分栏间隔

分栏之间存在间隔。



Row 组件 提供 `gutter` 属性来指定每一栏之间的间隔，默认间隔为 0。

```

1. <el-row :gutter="20">
2.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
   col>
3.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
   col>
4.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
   col>
5.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
   col>
6. </el-row>
7.
8. <style>
9.   .el-row {
10.     margin-bottom: 20px;
11.     &:last-child {
12.       margin-bottom: 0;
13.     }
14.   }
15.   .el-col {
16.     border-radius: 4px;
17.   }
18.   .bg-purple-dark {
19.     background: #99a9bf;
20.   }
21.   .bg-purple {
22.     background: #d3dce6;
23.   }
24.   .bg-purple-light {
25.     background: #e5e9f2;
26.   }
27.   .grid-content {
28.     border-radius: 4px;

```

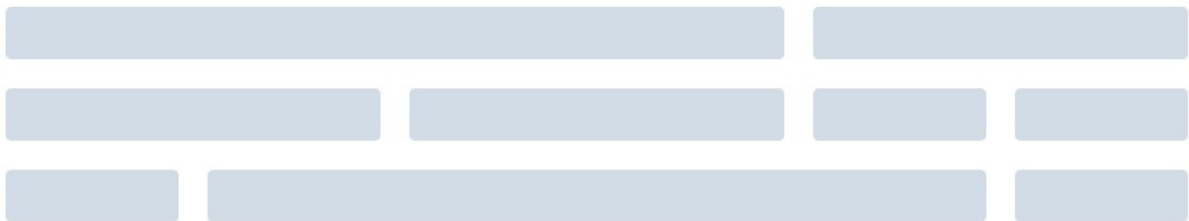
```

29.     min-height: 36px;
30.   }
31.   .row-bg {
32.     padding: 10px 0;
33.     background-color: #f9fafc;
34.   }
35. </style>

```

## 混合布局

通过基础的 1/24 分栏任意扩展组合形成较为复杂的混合布局。



```

1. <el-row :gutter="20">
2.   <el-col :span="16"><div class="grid-content bg-purple"></div>
   </el-col>
3.   <el-col :span="8"><div class="grid-content bg-purple"></div></el-
   col>
4. </el-row>
5. <el-row :gutter="20">
6.   <el-col :span="8"><div class="grid-content bg-purple"></div></el-
   col>
7.   <el-col :span="8"><div class="grid-content bg-purple"></div></el-
   col>
8.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
   col>
9.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
   col>
10. </el-row>
11. <el-row :gutter="20">
12.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
   col>
13.   <el-col :span="16"><div class="grid-content bg-purple"></div>

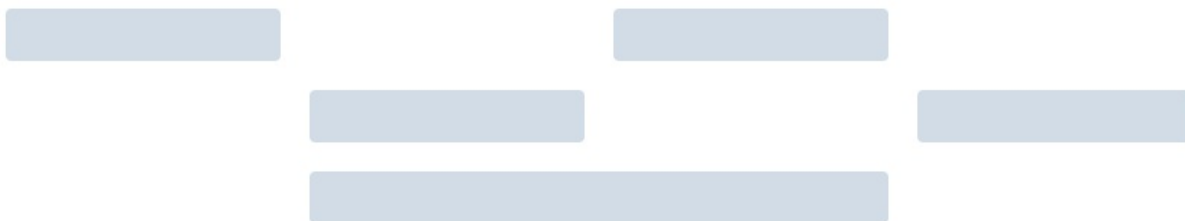
```

```
    </el-col>
14.   <el-col :span="4"><div class="grid-content bg-purple"></div></el-
    col>
15. </el-row>
16.
17. <style>
18.   .el-row {
19.     margin-bottom: 20px;
20.     &:last-child {
21.       margin-bottom: 0;
22.     }
23.   }
24.   .el-col {
25.     border-radius: 4px;
26.   }
27.   .bg-purple-dark {
28.     background: #99a9bf;
29.   }
30.   .bg-purple {
31.     background: #d3dce6;
32.   }
33.   .bg-purple-light {
34.     background: #e5e9f2;
35.   }
36.   .grid-content {
37.     border-radius: 4px;
38.     min-height: 36px;
39.   }
40.   .row-bg {
41.     padding: 10px 0;
42.     background-color: #f9fafc;
43.   }
44. </style>
```

## 分栏偏移

支持偏移指定的栏数。





通过制定 `col` 组件的 `offset` 属性可以指定分栏偏移的栏数。

```

1. <el-row :gutter="20">
2.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-
   col>
3.   <el-col :span="6" :offset="6"><div class="grid-content bg-
   purple"></div></el-col>
4. </el-row>
5. <el-row :gutter="20">
6.   <el-col :span="6" :offset="6"><div class="grid-content bg-
   purple"></div></el-col>
7.   <el-col :span="6" :offset="6"><div class="grid-content bg-
   purple"></div></el-col>
8. </el-row>
9. <el-row :gutter="20">
10.  <el-col :span="12" :offset="6"><div class="grid-content bg-
   purple"></div></el-col>
11. </el-row>
12.
13. <style>
14.   .el-row {
15.     margin-bottom: 20px;
16.     &:last-child {
17.       margin-bottom: 0;
18.     }
19.   }
20.   .el-col {
21.     border-radius: 4px;
22.   }
23.   .bg-purple-dark {
24.     background: #99a9bf;
25.   }

```

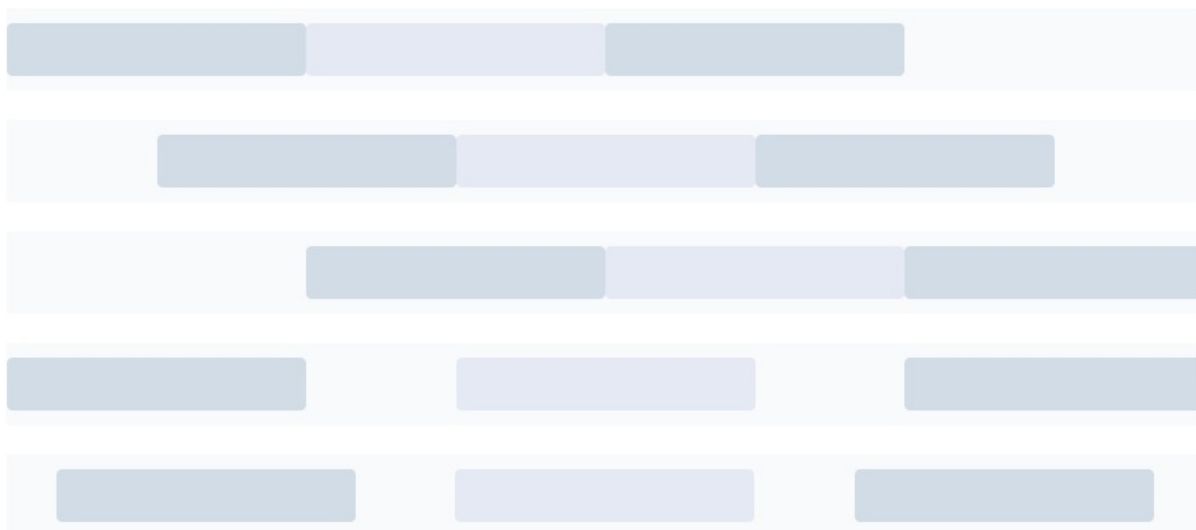
```

26.   .bg-purple {
27.     background: #d3dce6;
28.   }
29.   .bg-purple-light {
30.     background: #e5e9f2;
31.   }
32.   .grid-content {
33.     border-radius: 4px;
34.     min-height: 36px;
35.   }
36.   .row-bg {
37.     padding: 10px 0;
38.     background-color: #f9fafc;
39.   }
40. </style>

```

## 对齐方式

通过 flex 布局来对分栏进行灵活的对齐。



将 `type` 属性赋值为 'flex'，可以启用 flex 布局，并可通过 `justify` 属性来指定 start, center, end, space-between, space-around 其中的值来定义子元素的排版方式。

```
1. <el-row type="flex" class="row-bg">
```

```
2.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
3.   <el-col :span="6"><div class="grid-content bg-purple-light">  
    </div></el-col>  
4.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
5. </el-row>  
6. <el-row type="flex" class="row-bg" justify="center">  
7.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
8.   <el-col :span="6"><div class="grid-content bg-purple-light">  
    </div></el-col>  
9.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
10. </el-row>  
11. <el-row type="flex" class="row-bg" justify="end">  
12.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
13.   <el-col :span="6"><div class="grid-content bg-purple-light">  
    </div></el-col>  
14.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
15. </el-row>  
16. <el-row type="flex" class="row-bg" justify="space-between">  
17.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
18.   <el-col :span="6"><div class="grid-content bg-purple-light">  
    </div></el-col>  
19.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
20. </el-row>  
21. <el-row type="flex" class="row-bg" justify="space-around">  
22.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>  
23.   <el-col :span="6"><div class="grid-content bg-purple-light">  
    </div></el-col>  
24.   <el-col :span="6"><div class="grid-content bg-purple"></div></el-  
    col>
```

```
25. </el-row>
26.
27. <style>
28.   .el-row {
29.     margin-bottom: 20px;
30.     &:last-child {
31.       margin-bottom: 0;
32.     }
33.   }
34.   .el-col {
35.     border-radius: 4px;
36.   }
37.   .bg-purple-dark {
38.     background: #99a9bf;
39.   }
40.   .bg-purple {
41.     background: #d3dce6;
42.   }
43.   .bg-purple-light {
44.     background: #e5e9f2;
45.   }
46.   .grid-content {
47.     border-radius: 4px;
48.     min-height: 36px;
49.   }
50.   .row-bg {
51.     padding: 10px 0;
52.     background-color: #f9fafc;
53.   }
54. </style>
```

## 响应式布局

参照了 Bootstrap 的 响应式设计，预设了五个响应尺寸：xs、sm、md、lg 和 xl。



```

1. <el-row :gutter="10">
2.   <el-col :xs="8" :sm="6" :md="4" :lg="3" :xl="1"><div class="grid-
   content bg-purple"></div></el-col>
3.   <el-col :xs="4" :sm="6" :md="8" :lg="9" :xl="11"><div
   class="grid-content bg-purple-light"></div></el-col>
4.   <el-col :xs="4" :sm="6" :md="8" :lg="9" :xl="11"><div
   class="grid-content bg-purple"></div></el-col>
5.   <el-col :xs="8" :sm="6" :md="4" :lg="3" :xl="1"><div class="grid-
   content bg-purple-light"></div></el-col>
6. </el-row>
7.
8. <style>
9.   .el-col {
10.     border-radius: 4px;
11.   }
12.   .bg-purple-dark {
13.     background: #99a9bf;
14.   }
15.   .bg-purple {
16.     background: #d3dce6;
17.   }
18.   .bg-purple-light {
19.     background: #e5e9f2;
20.   }
21.   .grid-content {
22.     border-radius: 4px;
23.     min-height: 36px;
24.   }
25. </style>

```

## 基于断点的隐藏类

Element 额外提供了一系列类名，用于在某些条件下隐藏元素。这些类名可以添加在任何 DOM 元素或自定义组件上。如果需要，请自行引入以下文件：

```
1. import 'element-ui/lib/theme-chalk/display.css';
```

包含的类名及其含义为：

- `hidden-xs-only` - 当视口在 `xs` 尺寸时隐藏
- `hidden-sm-only` - 当视口在 `sm` 尺寸时隐藏
- `hidden-sm-and-down` - 当视口在 `sm` 及以下尺寸时隐藏
- `hidden-sm-and-up` - 当视口在 `sm` 及以上尺寸时隐藏
- `hidden-md-only` - 当视口在 `md` 尺寸时隐藏
- `hidden-md-and-down` - 当视口在 `md` 及以下尺寸时隐藏
- `hidden-md-and-up` - 当视口在 `md` 及以上尺寸时隐藏
- `hidden-lg-only` - 当视口在 `lg` 尺寸时隐藏
- `hidden-lg-and-down` - 当视口在 `lg` 及以下尺寸时隐藏
- `hidden-lg-and-up` - 当视口在 `lg` 及以上尺寸时隐藏
- `hidden-xl-only` - 当视口在 `xl` 尺寸时隐藏

## Row Attributes

参数	说明	类型	可选值	默认值
<code>gutter</code>	栅格间隔	number	—	0
<code>type</code>	布局模式，可选 flex，现代浏览器下有效	string	—	—
<code>justify</code>	flex 布局下的水平排列方式	string	start/end/center/space-around/space-between	start
<code>align</code>	flex 布局下的垂直排列方式	string	top/middle/bottom	top
<code>tag</code>	自定义元素标签	string	*	div

## Col Attributes

参数	说明	类型	可选值	默认值

span	栅格占据的列数	number	—	24
offset	栅格左侧的间隔格数	number	—	0
push	栅格向右移动格数	number	—	0
pull	栅格向左移动格数	number	—	0
xs	<768px 响应式栅格数或者栅格属性对象	number/object (例如: {span: 4, offset: 4})	—	—
sm	≥768px 响应式栅格数或者栅格属性对象	number/object (例如: {span: 4, offset: 4})	—	—
md	≥992px 响应式栅格数或者栅格属性对象	number/object (例如: {span: 4, offset: 4})	—	—
lg	≥1200px 响应式栅格数或者栅格属性对象	number/object (例如: {span: 4, offset: 4})	—	—
xl	≥1920px 响应式栅格数或者栅格属性对象	number/object (例如: {span: 4, offset: 4})	—	—
tag	自定义元素标签	string	*	div

## 3. 表单组件



## 3.1 Radio 单选框

- Radio 单选框
  - 基础用法
  - 禁用状态
  - 单选框组
  - 按钮样式
  - 带有边框
  - Radio Attributes
  - Radio Events
  - Radio-group Attributes
  - Radio-group Events
  - Radio-button Attributes

## Radio 单选框

在一组备选项中进行单选

## 基础用法

由于选项默认可见，不宜过多，若选项过多，建议使用 Select 选择器。

☒ 备选项    ☐ 备选项

要使用 `Radio` 组件，只需要设置 `v-model` 绑定变量，选中意味着变量的值为相应 `Radio label` 属性的值，`label` 可以是 `String`、`Number` 或 `Boolean`。

```
1. <template>
2.   <el-radio v-model="radio" label="1">备选项</el-radio>
```

```

3.   <el-radio v-model="radio" label="2">备选项</el-radio>
4. </template>
5.
6. <script>
7.   export default {
8.     data () {
9.       return {
10.        radio: '1'
11.      };
12.    }
13.  }
14. </script>

```

## 禁用状态

单选框不可用的状态。

☐ 备选项    ☒ 备选项

只要在 `el-radio` 元素中设置 `disabled` 属性即可，它接受一个 Boolean，`true` 为禁用。

```

1. <template>
2.   <el-radio disabled v-model="radio1" label="禁用">备选项</el-radio>
3.   <el-radio disabled v-model="radio1" label="选中且禁用">备选项</el-radio>
4. </template>
5.
6. <script>
7.   export default {
8.     data () {
9.       return {
10.        radio1: '选中且禁用'
11.      };
12.    }
13.  }

```

```
14. </script>
```

## 单选框组

适用于在多个互斥的选项中选择场景

☒ 备选项    ☐ 备选项    ☐ 备选项

结合 `el-radio-group` 元素和子元素 `el-radio` 可以实现单选组，在 `el-radio-group` 中绑定 `v-model`，在 `el-radio` 中设置好 `label` 即可，无需再给每一个 `el-radio` 绑定变量，另外，还提供了 `change` 事件来响应变化，它会传入一个参数 `value`。

```
1. <template>
2.   <el-radio-group v-model="radio2">
3.     <el-radio :label="3">备选项</el-radio>
4.     <el-radio :label="6">备选项</el-radio>
5.     <el-radio :label="9">备选项</el-radio>
6.   </el-radio-group>
7. </template>
8.
9. <script>
10.   export default {
11.     data () {
12.       return {
13.         radio2: 3
14.       };
15.     }
16.   }
17. </script>
```

## 按钮样式

按钮样式的单选组合。



只需要把 `el-radio` 元素换成 `el-radio-button` 元素即可，此外，`Element` 还提供了 `size` 属性。

```

1. <template>
2.   <div>
3.     <el-radio-group v-model="radio3">
4.       <el-radio-button label="上海"></el-radio-button>
5.       <el-radio-button label="北京"></el-radio-button>
6.       <el-radio-button label="广州"></el-radio-button>
7.       <el-radio-button label="深圳"></el-radio-button>
8.     </el-radio-group>
9.   </div>
10.  <div style="margin-top: 20px">
11.    <el-radio-group v-model="radio4" size="medium">
12.      <el-radio-button label="上海" ></el-radio-button>
13.      <el-radio-button label="北京"></el-radio-button>
14.      <el-radio-button label="广州"></el-radio-button>
15.      <el-radio-button label="深圳"></el-radio-button>
16.    </el-radio-group>
17.  </div>
18.  <div style="margin-top: 20px">
19.    <el-radio-group v-model="radio5" size="small">
20.      <el-radio-button label="上海"></el-radio-button>
21.      <el-radio-button label="北京" disabled ></el-radio-button>
22.      <el-radio-button label="广州"></el-radio-button>
23.      <el-radio-button label="深圳"></el-radio-button>
24.    </el-radio-group>
25.  </div>

```

```

26.   <div style="margin-top: 20px">
27.     <el-radio-group v-model="radio6" disabled size="mini">
28.       <el-radio-button label="上海"></el-radio-button>
29.       <el-radio-button label="北京"></el-radio-button>
30.       <el-radio-button label="广州"></el-radio-button>
31.       <el-radio-button label="深圳"></el-radio-button>
32.     </el-radio-group>
33.   </div>
34. </template>
35.
36. <script>
37.   export default {
38.     data () {
39.       return {
40.         radio3: '上海',
41.         radio4: '上海',
42.         radio5: '上海',
43.         radio6: '上海'
44.       };
45.     }
46.   }
47. </script>

```

## 带有边框

☒ 备选项1
 ☐ 备选项2

☒ 备选项1
 ☐ 备选项2

☒ 备选项1
 ☐ 备选项2

☐ 备选项1
 ☐ 备选项2

设置 `border` 属性可以渲染为带有边框的单选框。

```

1. <template>
2.   <div>
3.     <el-radio v-model="radio7" label="1" border>备选项1</el-radio>
4.     <el-radio v-model="radio7" label="2" border>备选项2</el-radio>
5.   </div>
6.   <div style="margin-top: 20px">
7.     <el-radio v-model="radio8" label="1" border size="medium">备选项
8.       1</el-radio>
9.     <el-radio v-model="radio8" label="2" border size="medium">备选项
10.      2</el-radio>
11.   </div>
12.   <div style="margin-top: 20px">
13.     <el-radio-group v-model="radio9" size="small">
14.       <el-radio label="1" border>备选项1</el-radio>
15.       <el-radio label="2" border disabled>备选项2</el-radio>
16.     </el-radio-group>
17.   </div>
18.   <div style="margin-top: 20px">
19.     <el-radio-group v-model="radio10" size="mini" disabled>
20.       <el-radio label="1" border>备选项1</el-radio>
21.       <el-radio label="2" border>备选项2</el-radio>
22.     </el-radio-group>
23.   </div>
24. </template>
25.
26. <script>
27.   export default {
28.     data () {
29.       return {
30.         radio7: '1',
31.         radio8: '1',
32.         radio9: '1',
33.         radio10: '1'
34.       };
35.     }
36.   }
37. </script>

```

## Radio Attributes

参数	说明	类型	可选值	默认值
label	Radio 的 value	string / number / boolean	—	—
disabled	是否禁用	boolean	—	false
border	是否显示边框	boolean	—	false
size	Radio 的尺寸, 仅在 border 为真时有效	string	medium / small / mini	—
name	原生 name 属性	string	—	—

## Radio Events

事件名称	说明	回调参数
change	绑定值变化时触发的事件	选中的 Radio label 值

## Radio-group Attributes

参数	说明	类型	可选值	默认值
size	单选框组尺寸, 仅对按钮形式的 Radio 或带有边框的 Radio 有效	string	medium / small / mini	—
disabled	是否禁用	boolean	—	false
text-color	按钮形式的 Radio 激活时的文本颜色	string	—	#ffffff
fill	按钮形式的 Radio 激活时的填充色和边框色	string	—	#409EFF

## Radio-group Events

事件名称	说明	回调参数
change	绑定值变化时触发的事件	选中的 Radio label 值

## Radio-button Attributes

--	--	--	--	--

参数	说明	类型	可选值	默认值
label	Radio 的 value	string / number	—	—
disabled	是否禁用	boolean	—	false
name	原生 name 属性	string	—	—



## 3.2 Checkbox 多选框

- [Checkbox 多选框](#)
  - [基础用法](#)
  - [禁用状态](#)
  - [多选框组](#)
  - [indeterminate 状态](#)
  - [可选项目数量的限制](#)
  - [按钮样式](#)
  - [带有边框](#)
  - [Checkbox Attributes](#)
  - [Checkbox Events](#)
  - [Checkbox-group Attributes](#)
  - [Checkbox-group Events](#)
  - [Checkbox-button Attributes](#)

## Checkbox 多选框

一组备选项中进行多选

### 基础用法

单独使用可以表示两种状态之间的切换，写在标签中的内容为 checkbox 按钮后的介绍。

☒ 备选项

在 `el-checkbox` 元素中定义 `v-model` 绑定变量，单一的 `checkbox` 中，默认绑定变量的值会是 `Boolean`，选中为 `true`。

```
1. <template>
2.   <!-- `checked` 为 true 或 false -->
3.   <el-checkbox v-model="checked">备选项</el-checkbox>
4. </template>
5. <script>
6.   export default {
7.     data() {
8.       return {
9.         checked: true
10.      };
11.    }
12.  };
13. </script>
```

## 禁用状态

多选框不可用状态。

☐ 备选项1    ☒ 备选项

设置disabled属性即可。

```
1. <template>
2.   <el-checkbox v-model="checked1" disabled>备选项1</el-checkbox>
3.   <el-checkbox v-model="checked2" disabled>备选项</el-checkbox>
4. </template>
5. <script>
6.   export default {
7.     data() {
8.       return {
9.         checked1: false,
10.        checked2: true
11.      };
12.    }
13.  };
14. </script>
```

## 多选框组

适用于多个勾选框绑定到同一个数组的情景，通过是否勾选来表示这一组选项中选中的项。

☒ 复选框 A    ☐ 复选框 B    ☐ 复选框 C    ☐ 禁用    ☒ 选中且禁用

`checkbox-group` 元素能把多个 `checkbox` 管理为一组，只需要在 `Group` 中使用 `v-model` 绑定Array类型的变量即可。 `el-checkbox` 的 `label` 属性是该 `checkbox` 对应的值，若该标签中无内容，则该属性也充当 `checkbox` 按钮后的介绍。 `label` 与数组中的元素值相对应，如果存在指定的值则为选中状态，否则为不选中。

```

1. <template>
2.   <el-checkbox-group v-model="checkList">
3.     <el-checkbox label="复选框 A"></el-checkbox>
4.     <el-checkbox label="复选框 B"></el-checkbox>
5.     <el-checkbox label="复选框 C"></el-checkbox>
6.     <el-checkbox label="禁用" disabled></el-checkbox>
7.     <el-checkbox label="选中且禁用" disabled></el-checkbox>
8.   </el-checkbox-group>
9. </template>
10.
11. <script>
12.   export default {
13.     data () {
14.       return {
15.         checkList: ['选中且禁用', '复选框 A']
16.       };
17.     }
18.   };
19. </script>

```

## indeterminate 状态

`indeterminate` 属性用以表示 checkbox 的不确定状态，一般用于实现全选的效果。

☐ 全选

☒ 上海 ☒ 北京 ☐ 广州 ☐ 深圳

```

1. <template>
2.   <el-checkbox :indeterminate="isIndeterminate" v-model="checkAll"
   @change="handleCheckAllChange">全选</el-checkbox>
3.   <div style="margin: 15px 0;"></div>
4.   <el-checkbox-group v-model="checkedCities"
   @change="handleCheckedCitiesChange">
5.     <el-checkbox v-for="city in cities" :label="city" :key="city">
       {{city}}</el-checkbox>
6.   </el-checkbox-group>
7. </template>
8. <script>
9.   const cityOptions = ['上海', '北京', '广州', '深圳'];
10.  export default {
11.    data() {
12.      return {
13.        checkAll: false,
14.        checkedCities: ['上海', '北京'],
15.        cities: cityOptions,
16.        isIndeterminate: true
17.      };
18.    },
19.    methods: {
20.      handleCheckAllChange(val) {
21.        this.checkedCities = val ? cityOptions : [];
22.        this.isIndeterminate = false;
23.      },
24.      handleCheckedCitiesChange(value) {
25.        let checkedCount = value.length;
26.        this.checkAll = checkedCount === this.cities.length;
27.        this.isIndeterminate = checkedCount > 0 && checkedCount <
           this.cities.length;

```

```
28.     }  
29.     }  
30.   };  
31. </script>
```

## 可选项目数量的限制

使用 `min` 和 `max` 属性能够限制可以被勾选的项目的数量。

☒ 上海 ☒ 北京 ☐ 广州 ☐ 深圳

```
1. <template>  
2.   <el-checkbox-group  
3.     v-model="checkedCities1"  
4.     :min="1"  
5.     :max="2">  
6.     <el-checkbox v-for="city in cities" :label="city" :key="city">  
       {{city}}</el-checkbox>  
7.   </el-checkbox-group>  
8. </template>  
9. <script>  
10.   const cityOptions = ['上海', '北京', '广州', '深圳'];  
11.   export default {  
12.     data() {  
13.       return {  
14.         checkedCities1: ['上海', '北京'],  
15.         cities: cityOptions  
16.       };  
17.     }  
18.   };  
19. </script>
```

## 按钮样式

按钮样式的多选组合。



只需要把 `el-checkbox` 元素替换为 `el-checkbox-button` 元素即可。此外，Element 还提供了 `size` 属性。

```

1. <template>
2.   <div>
3.     <el-checkbox-group v-model="checkboxGroup1">
4.       <el-checkbox-button v-for="city in cities" :label="city"
       :key="city">{{city}}</el-checkbox-button>
5.     </el-checkbox-group>
6.   </div>
7.   <div style="margin-top: 20px">
8.     <el-checkbox-group v-model="checkboxGroup2" size="medium">
9.       <el-checkbox-button v-for="city in cities" :label="city"
       :key="city">{{city}}</el-checkbox-button>
10.    </el-checkbox-group>
11.  </div>
12.  <div style="margin-top: 20px">
13.    <el-checkbox-group v-model="checkboxGroup3" size="small">
14.      <el-checkbox-button v-for="city in cities" :label="city"
       :disabled="city === '北京'" :key="city">{{city}}</el-checkbox-
       button>
15.    </el-checkbox-group>
16.  </div>
17.  <div style="margin-top: 20px">
18.    <el-checkbox-group v-model="checkboxGroup4" size="mini"
       disabled>
19.      <el-checkbox-button v-for="city in cities" :label="city"
       :key="city">{{city}}</el-checkbox-button>
20.    </el-checkbox-group>

```

```

21.   </div>
22. </template>
23. <script>
24.   const cityOptions = ['上海', '北京', '广州', '深圳'];
25.   export default {
26.     data () {
27.       return {
28.         checkboxGroup1: ['上海'],
29.         checkboxGroup2: ['上海'],
30.         checkboxGroup3: ['上海'],
31.         checkboxGroup4: ['上海'],
32.         cities: cityOptions
33.       };
34.     }
35.   }
36. </script>

```

## 带有边框

The image displays four distinct visual states of a checkbox group with a border attribute. Each state consists of two adjacent rectangular buttons, each containing a checkbox icon and a label. The labels are '备选项1' (Option 1) and '备选项2' (Option 2).  
 - The first state shows '备选项1' with a blue checkmark and '备选项2' with an empty white checkbox.  
 - The second state shows '备选项1' with an empty white checkbox and '备选项2' with a blue checkmark.  
 - The third state shows both '备选项1' and '备选项2' with empty white checkboxes.  
 - The fourth state shows both '备选项1' and '备选项2' with light blue disabled checkboxes.

设置 `border` 属性可以渲染为带有边框的多选框。

```

1. <template>
2.   <div>
3.     <el-checkbox v-model="checked3" label="备选项1" border></el-
checkbox>
4.     <el-checkbox v-model="checked4" label="备选项2" border></el-
checkbox>

```

```

5.     </div>
6.     <div style="margin-top: 20px">
7.         <el-checkbox v-model="checked5" label="备选项1" border
size="medium"></el-checkbox>
8.         <el-checkbox v-model="checked6" label="备选项2" border
size="medium"></el-checkbox>
9.     </div>
10.    <div style="margin-top: 20px">
11.        <el-checkbox-group v-model="checkboxGroup5" size="small">
12.            <el-checkbox label="备选项1" border></el-checkbox>
13.            <el-checkbox label="备选项2" border disabled></el-checkbox>
14.        </el-checkbox-group>
15.    </div>
16.    <div style="margin-top: 20px">
17.        <el-checkbox-group v-model="checkboxGroup6" size="mini"
disabled>
18.            <el-checkbox label="备选项1" border></el-checkbox>
19.            <el-checkbox label="备选项2" border></el-checkbox>
20.        </el-checkbox-group>
21.    </div>
22. </template>
23.
24. <script>
25.     export default {
26.         data () {
27.             return {
28.                 checked3: true,
29.                 checked4: false,
30.                 checked5: false,
31.                 checked6: true,
32.                 checkboxGroup5: [],
33.                 checkboxGroup6: []
34.             };
35.         }
36.     }
37. </script>

```



## Checkbox Attributes

参数	说明	类型	可选值	默认值
label	选中状态的值（只有在 checkbox-group 或者绑定对象类型为 array 时有效）	string / number / boolean	—	—
true-label	选中时的值	string / number	—	—
false-label	没有选中时的值	string / number	—	—
disabled	是否禁用	boolean	—	false
border	是否显示边框	boolean	—	false
size	Checkbox 的尺寸，仅在 border 为真时有效	string	medium / small / mini	—
name	原生 name 属性	string	—	—
checked	当前是否勾选	boolean	—	false
indeterminate	设置 indeterminate 状态，只负责样式控制	boolean	—	false

## Checkbox Events

事件名称	说明	回调参数
change	当绑定值变化时触发的事件	更新后的值

## Checkbox-group Attributes

参数	说明	类型	可选值	默认值
size	多选框组尺寸，仅对按钮形式的 Checkbox 或带有边框的 Checkbox 有效	string	medium / small / mini	—
disabled	是否禁用	boolean	—	false
min	可被勾选的 checkbox 的最小数量	number	—	—
max	可被勾选的 checkbox 的最大数量	number	—	—
text-color	按钮形式的 Checkbox 激活时的文本颜色	string	—	#ffffff
fill	按钮形式的 Checkbox 激活时的填充色和边框色	string	—	#409EFF

## Checkbox-group Events

事件名称	说明	回调参数
change	当绑定值变化时触发的事件	更新后的值

## Checkbox-button Attributes

参数	说明	类型	可选值	默认值
label	选中状态的值（只有在checkbox-group或者绑定对象类型为array时有效）	string / number / boolean	—	—
true-label	选中时的值	string / number	—	—
false-label	没有选中时的值	string / number	—	—
disabled	是否禁用	boolean	—	false
name	原生 name 属性	string	—	—
checked	当前是否勾选	boolean	—	false

## 3.3 Input 输入框

- Input 输入框
  - 基础用法
  - 禁用状态
  - 可清空
  - 带 icon 的输入框
  - 文本域
  - 可自适应文本高度的文本域
  - 复合型输入框
  - 尺寸
  - 带输入建议
  - 自定义模板
  - 远程搜索
  - Input Attributes
  - Input slots
  - Input Events
  - Input Methods
  - Autocomplete Attributes
  - Autocomplete slots
  - Autocomplete Events

## Input 输入框

---

通过鼠标或键盘输入字符

### 基础用法

---

```

1. <el-input v-model="input" placeholder="请输入内容"></el-input>
2.
3. <script>
4.   export default {
5.     data() {
6.       return {
7.         input: ''
8.       }
9.     }
10.  }
11. </script>

```

## 禁用状态

通过 `disabled` 属性指定是否禁用 input 组件

```

1. <el-input
2.   placeholder="请输入内容"
3.   v-model="input1"
4.   :disabled="true">
5. </el-input>
6.
7. <script>
8.   export default {
9.     data() {
10.      return {
11.        input1: ''
12.      }
13.    }

```

```

14.   }
15. </script>

```

## 可清空

使用 `clearable` 属性即可得到一个可清空的输入框

```

1. <el-input
2.   placeholder="请输入内容"
3.   v-model="input10"
4.   clearable>
5. </el-input>
6.
7. <script>
8.   export default {
9.     data() {
10.      return {
11.        input10: ''
12.      }
13.    }
14.  }
15. </script>

```

## 带 icon 的输入框

带有图标标记输入类型

属性方式：




slot 方式：



可以通过 `prefix-icon` 和 `suffix-icon` 属性在 `input` 组件头部和尾部增加显示图标，也可以通过 `slot` 来放置图标。

```

1. <div class="demo-input-suffix">
2.   属性方式：
3.   <el-input
4.     placeholder="请选择日期"
5.     suffix-icon="el-icon-date"
6.     v-model="input2">
7.   </el-input>
8.   <el-input
9.     placeholder="请输入内容"
10.    prefix-icon="el-icon-search"
11.    v-model="input21">
12.   </el-input>
13. </div>
14. <div class="demo-input-suffix">
15.   slot 方式：
16.   <el-input
17.     placeholder="请选择日期"
18.     v-model="input22">
19.     <i slot="suffix" class="el-input__icon el-icon-date"></i>
20.   </el-input>
21.   <el-input
22.     placeholder="请输入内容"
23.     v-model="input23">
24.     <i slot="prefix" class="el-input__icon el-icon-search"></i>
25.   </el-input>
26. </div>
27.
28. <script>
29. export default {
30.   data() {
31.     return {
32.       input2: '',
33.       input21: '',
34.       input22: '',
35.       input23: ''

```

```
36.     }  
37.   }  
38. }  
39. </script>
```

## 文本域

用于输入多行文本信息，通过将 `type` 属性的值指定为 `textarea`。

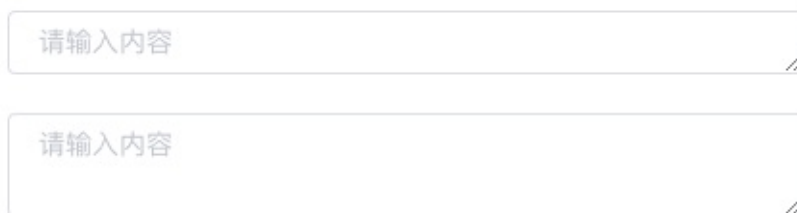
文本域高度可通过 `rows` 属性控制

```
1. <el-input  
2.   type="textarea"  
3.   :rows="2"  
4.   placeholder="请输入内容"  
5.   v-model="textarea">  
6. </el-input>  
7.  
8. <script>  
9. export default {  
10.   data() {  
11.     return {  
12.       textarea: ''  
13.     }  
14.   }  
15. }  
16. </script>
```

## 可自适应文本高度的文本域

通过设置 `autosize` 属性可以使得文本域的高度能够根据文本内容自

动进行调整，并且 `autosize` 还可以设定为一个对象，指定最小行数和最大行数。



```

1. <el-input
2.   type="textarea"
3.   autosize
4.   placeholder="请输入内容"
5.   v-model="textarea2">
6. </el-input>
7. <div style="margin: 20px 0;"></div>
8. <el-input
9.   type="textarea"
10.  :autosize="{ minRows: 2, maxRows: 4}"
11.  placeholder="请输入内容"
12.  v-model="textarea3">
13. </el-input>
14.
15. <script>
16. export default {
17.   data() {
18.     return {
19.       textarea2: '',
20.       textarea3: ''
21.     }
22.   }
23. }
24. </script>

```

## 复合型输入框



可前置或后置元素，一般为标签或按钮

The image displays three variations of the `el-input` component. The first example shows a text input with 'Http://' prepended to the left. The second example shows a text input with '.com' appended to the right. The third example shows a more complex input with a dropdown menu (placeholder '请选择') prepended to the left and a search button (placeholder '请输入内容' with a magnifying glass icon) appended to the right.

可通过 `slot` 来指定在 `input` 中前置或者后置内容。

```

1. <div>
2.   <el-input placeholder="请输入内容" v-model="input3">
3.     <template slot="prepend">Http://</template>
4.   </el-input>
5. </div>
6. <div style="margin-top: 15px;">
7.   <el-input placeholder="请输入内容" v-model="input4">
8.     <template slot="append">.com</template>
9.   </el-input>
10. </div>
11. <div style="margin-top: 15px;">
12.   <el-input placeholder="请输入内容" v-model="input5" class="input-
    with-select">
13.     <el-select v-model="select" slot="prepend" placeholder="请选择">
14.       <el-option label="餐厅名" value="1"></el-option>
15.       <el-option label="订单号" value="2"></el-option>
16.       <el-option label="用户电话" value="3"></el-option>
17.     </el-select>
18.     <el-button slot="append" icon="el-icon-search"></el-button>
19.   </el-input>
20. </div>
21. <style>
22.   .el-select .el-input {
23.     width: 130px;
24.   }
25.   .input-with-select .el-input-group__prepend {
26.     background-color: #fff;
27.   }

```

```

28. </style>
29. <script>
30. export default {
31.   data() {
32.     return {
33.       input3: '',
34.       input4: '',
35.       input5: '',
36.       select: ''
37.     }
38.   }
39. }
40. </script>

```

## 尺寸



可通过 `size` 属性指定输入框的尺寸，除了默认的大小外，还提供了 `large`、`small` 和 `mini` 三种尺寸。

```

1. <div class="demo-input-size">
2.   <el-input
3.     placeholder="请输入内容"
4.     suffix-icon="el-icon-date"
5.     v-model="input6">
6. </el-input>
7. <el-input
8.   size="medium"
9.   placeholder="请输入内容"
10.  suffix-icon="el-icon-date"
11.  v-model="input7">
12. </el-input>
13. <el-input
14.   size="small"
15.   placeholder="请输入内容"

```

```

16.     suffix-icon="el-icon-date"
17.     v-model="input8">
18.   </el-input>
19.   <el-input
20.     size="mini"
21.     placeholder="请输入内容"
22.     suffix-icon="el-icon-date"
23.     v-model="input9">
24.   </el-input>
25. </div>
26.
27. <script>
28. export default {
29.   data() {
30.     return {
31.       input6: '',
32.       input7: '',
33.       input8: '',
34.       input9: ''
35.     }
36.   }
37. }
38. </script>

```

## 带输入建议

根据输入内容提供对应的输入建议

激活即列出输入建议

请输入内容

输入后匹配输入建议

请输入内容

`autocomplete` 是一个可带输入建议的输入框组件，`fetch-`

`suggestions` 是一个返回输入建议的方法属性，如

`querySearch(queryString, cb)`，在该方法中你可以在你的输入建议数据准备好时通过 `cb(data)` 返回到 `autocomplete` 组件

中。

```

1. <el-row class="demo-autocomplete">
2.   <el-col :span="12">
3.     <div class="sub-title">激活即列出输入建议</div>
4.     <el-autocomplete
5.       class="inline-input"
6.       v-model="state1"
7.       :fetch-suggestions="querySearch"
8.       placeholder="请输入内容"
9.       @select="handleSelect"
10.    ></el-autocomplete>
11.  </el-col>
12.  <el-col :span="12">
13.    <div class="sub-title">输入后匹配输入建议</div>
14.    <el-autocomplete
15.      class="inline-input"
16.      v-model="state2"
17.      :fetch-suggestions="querySearch"
18.      placeholder="请输入内容"
19.      :trigger-on-focus="false"
20.      @select="handleSelect"
21.    ></el-autocomplete>
22.  </el-col>
23. </el-row>
24. <script>
25.   export default {
26.     data() {
27.       return {
28.         restaurants: [],
29.         state1: '',
30.         state2: ''
31.       };
32.     },
33.     methods: {
34.       querySearch(queryString, cb) {
35.         var restaurants = this.restaurants;
36.         var results = queryString ?

```

```

    restaurants.filter(this.createFilter(queryString)) : restaurants;
37.      // 调用 callback 返回建议列表的数据
38.      cb(results);
39.    },
40.    createFilter(queryString) {
41.      return (restaurant) => {
42.        return
        (restaurant.value.toLowerCase().indexOf(queryString.toLowerCase())
        === 0);
43.      };
44.    },
45.    loadAll() {
46.      return [
47.        { "value": "三全鲜食（北新泾店）", "address": "长宁区新渔路144
        号" },
48.        { "value": "Hot honey 首尔炸鸡（仙霞路）", "address": "上海市
        长宁区淞虹路661号" },
49.        { "value": "新旺角茶餐厅", "address": "上海市普陀区真北路988号
        创邑金沙谷6号楼113" },
50.        { "value": "泷千家（天山西路店）", "address": "天山西路438号"
        },
51.        { "value": "胖仙女纸杯蛋糕（上海凌空店）", "address": "上海市长
        宁区金钟路968号1幢18号楼一层商铺18-101" },
52.        { "value": "贡茶", "address": "上海市长宁区金钟路633号" },
53.        { "value": "豪大大香鸡排超级奶爸", "address": "上海市嘉定区曹安
        公路曹安路1685号" },
54.        { "value": "茶芝兰（奶茶，手抓饼）", "address": "上海市普陀区同
        普路1435号" },
55.        { "value": "十二泷町", "address": "上海市北翟路1444弄81号B
        幢-107" },
56.        { "value": "星移浓缩咖啡", "address": "上海市嘉定区新郁路817
        号" },
57.        { "value": "阿姨奶茶/豪大大", "address": "嘉定区曹安路1611号"
        },
58.        { "value": "新麦甜四季甜品炸鸡", "address": "嘉定区曹安公路
        2383弄55号" },
59.        { "value": "Monica摩托主题咖啡店", "address": "嘉定区江桥镇曹
        安公路2409号1F, 2383弄62号1F" },

```

```

60.      { "value": "浮生若茶（凌空soho店）", "address": "上海长宁区金
        钟路968号9号楼地下一层" },
61.      { "value": "NONO JUICE 鲜榨果汁", "address": "上海市长宁区天
        山西路119号" },
62.      { "value": "CoCo都可（北新泾店）", "address": "上海市长宁区仙霞
        西路" },
63.      { "value": "快乐柠檬（神州智慧店）", "address": "上海市长宁区天
        山西路567号1层R117号店铺" },
64.      { "value": "Merci Paul cafe", "address": "上海市普陀区光复西
        路丹巴路28弄6号楼819" },
65.      { "value": "猫山王（西郊百联店）", "address": "上海市长宁区仙霞
        西路88号第一层G05-F01-1-306" },
66.      { "value": "枪会山", "address": "上海市普陀区棕榈路" },
67.      { "value": "纵食", "address": "元丰天山花园（东门） 双流路267号"
        },
68.      { "value": "钱记", "address": "上海市长宁区天山西路" },
69.      { "value": "壹杯加", "address": "上海市长宁区通协路" },
70.      { "value": "吵哇嘀咖", "address": "上海市长宁区新泾镇金钟路999
        号2幢（B幢）第01层第1-02A单元" },
71.      { "value": "爱茜茜里（西郊百联）", "address": "长宁区仙霞西路88
        号1305室" },
72.      { "value": "爱茜茜里（近铁广场）", "address": "上海市普陀区真北路
        818号近铁城市广场北区地下二楼N-B2-02-C商铺" },
73.      { "value": "鲜果榨汁（金沙江路和美广店）", "address": "普陀区金
        沙江路2239号金沙和美广场B1-10-6" },
74.      { "value": "开心丽果（缤谷店）", "address": "上海市长宁区威宁路
        天山路341号" },
75.      { "value": "超级鸡车（丰庄路店）", "address": "上海市嘉定区丰庄
        路240号" },
76.      { "value": "妙生活果园（北新泾店）", "address": "长宁区新渔路
        144号" },
77.      { "value": "香宜度麻辣香锅", "address": "长宁区淞虹路148号" },
78.      { "value": "凡仔汉堡（老真北路店）", "address": "上海市普陀区老
        真北路160号" },
79.      { "value": "港式小铺", "address": "上海市长宁区金钟路968号15楼
        15-105室" },
80.      { "value": "蜀香源麻辣香锅（剑河路店）", "address": "剑河路
        443-1" },

```

```

81.         { "value": "北京饺子馆", "address": "长宁区北新泾街道天山西路
            490-1号" },
82.         { "value": "饭典*新简餐（凌空SOHO店）", "address": "上海市长宁
            区金钟路968号9号楼地下一层9-83室" },
83.         { "value": "焦耳·川式快餐（金钟路店）", "address": "上海市金钟
            路633号地下一层甲部" },
84.         { "value": "动力鸡车", "address": "长宁区仙霞西路299弄3号
            101B" },
85.         { "value": "浏阳蒸菜", "address": "天山西路430号" },
86.         { "value": "四海游龙（天山西路店）", "address": "上海市长宁区天
            山西路" },
87.         { "value": "樱花食堂（凌空店）", "address": "上海市长宁区金钟路
            968号15楼15-105室" },
88.         { "value": "壹分米客家传统调制米粉(天山店)", "address": "天山西
            路428号" },
89.         { "value": "福荣祥烧腊（平溪路店）", "address": "上海市长宁区协
            和路福泉路255弄57-73号" },
90.         { "value": "速记黄焖鸡米饭", "address": "上海市长宁区北新泾街道
            金钟路180号1层01号摊位" },
91.         { "value": "红辣椒麻辣烫", "address": "上海市长宁区天山西路492
            号" },
92.         { "value": "(小杨生煎)西郊百联餐厅", "address": "长宁区仙霞西路
            88号百联2楼" },
93.         { "value": "阳阳麻辣烫", "address": "天山西路389号" },
94.         { "value": "南拳妈妈龙虾盖浇饭", "address": "普陀区金沙江路
            1699号鑫乐惠美食广场A13" }
95.     ];
96. },
97.     handleSelect(item) {
98.         console.log(item);
99.     }
100. },
101.     mounted() {
102.         this.restaurants = this.loadAll();
103.     }
104. }
105. </script>

```

## 自定义模板

### 可自定义输入建议的显示



使用 `scoped slot` 自定义输入建议的模板。该 `scope` 的参数为 `item`，表示当前输入建议对象。

```

1. <el-autocomplete
2.   popper-class="my-autocomplete"
3.   v-model="state3"
4.   :fetch-suggestions="querySearch"
5.   placeholder="请输入内容"
6.   @select="handleSelect">
7.     <i
8.       class="el-icon-edit el-input__icon"
9.       slot="suffix"
10.      @click="handleIconClick">
11.    </i>
12.    <template slot-scope="props">
13.      <div class="name">{{ props.item.value }}</div>
14.      <span class="addr">{{ props.item.address }}</span>
15.    </template>
16.  </el-autocomplete>

```



```
17.
18. <style>
19. .my-autocomplete {
20.   li {
21.     line-height: normal;
22.     padding: 7px;
23.
24.     .name {
25.       text-overflow: ellipsis;
26.       overflow: hidden;
27.     }
28.     .addr {
29.       font-size: 12px;
30.       color: #b4b4b4;
31.     }
32.
33.     .highlighted .addr {
34.       color: #ddd;
35.     }
36.   }
37. }
38. </style>
39.
40. <script>
41.   export default {
42.     data() {
43.       return {
44.         restaurants: [],
45.         state3: ''
46.       };
47.     },
48.     methods: {
49.       querySearch(queryString, cb) {
50.         var restaurants = this.restaurants;
51.         var results = queryString ?
52.           restaurants.filter(this.createFilter(queryString)) : restaurants;
53.         // 调用 callback 返回建议列表的数据
54.         cb(results);
```

```

54.     },
55.     createFilter(queryString) {
56.         return (restaurant) => {
57.             return
126         (restaurant.value.toLowerCase().indexOf(queryString.toLowerCase())
127         === 0);
58.         };
59.     },
60.     loadAll() {
61.         return [
62.             { "value": "三全鲜食（北新泾店）", "address": "长宁区新渔路144
128             号" },
63.             { "value": "Hot honey 首尔炸鸡（仙霞路）", "address": "上海市
129             长宁区淞虹路661号" },
64.             { "value": "新旺角茶餐厅", "address": "上海市普陀区真北路988号
130             创邑金沙谷6号楼113" },
65.             { "value": "泷千家（天山西路店）", "address": "天山西路438号"
131             },
66.             { "value": "胖仙女纸杯蛋糕（上海凌空店）", "address": "上海市长
132             宁区金钟路968号1幢18号楼一层商铺18-101" },
67.             { "value": "贡茶", "address": "上海市长宁区金钟路633号" },
68.             { "value": "豪大大香鸡排超级奶爸", "address": "上海市嘉定区曹安
133             公路曹安路1685号" },
69.             { "value": "茶芝兰（奶茶，手抓饼）", "address": "上海市普陀区同
134             普路1435号" },
70.             { "value": "十二泷町", "address": "上海市北翟路1444弄81号B
135             幢-107" },
71.             { "value": "星移浓缩咖啡", "address": "上海市嘉定区新郁路817
136             号" },
72.             { "value": "阿姨奶茶/豪大大", "address": "嘉定区曹安路1611号"
137             },
73.             { "value": "新麦甜四季甜品炸鸡", "address": "嘉定区曹安公路
138             2383弄55号" },
74.             { "value": "Monica摩托主题咖啡店", "address": "嘉定区江桥镇曹
139             安公路2409号1F, 2383弄62号1F" },
75.             { "value": "浮生若茶（凌空soho店）", "address": "上海长宁区金
140             钟路968号9号楼地下一层" },
76.             { "value": "NONO JUICE 鲜榨果汁", "address": "上海市长宁区天

```

```

    山西路119号" },
77.      { "value": "CoCo都可(北新泾店)", "address": "上海市长宁区仙霞
    西路" },
78.      { "value": "快乐柠檬(神州智慧店)", "address": "上海市长宁区天
    山西路567号1层R117号店铺" },
79.      { "value": "Merci Paul cafe", "address": "上海市普陀区光复西
    路丹巴路28弄6号楼819" },
80.      { "value": "猫山王(西郊百联店)", "address": "上海市长宁区仙霞
    西路88号第一层G05-F01-1-306" },
81.      { "value": "枪会山", "address": "上海市普陀区棕榈路" },
82.      { "value": "纵食", "address": "元丰天山花园(东门) 双流路267号"
    },
83.      { "value": "钱记", "address": "上海市长宁区天山西路" },
84.      { "value": "壹杯加", "address": "上海市长宁区通协路" },
85.      { "value": "吵哇嘀咖", "address": "上海市长宁区新泾镇金钟路999
    号2幢(B幢)第01层第1-02A单元" },
86.      { "value": "爱茜茜里(西郊百联)", "address": "长宁区仙霞西路88
    号1305室" },
87.      { "value": "爱茜茜里(近铁广场)", "address": "上海市普陀区真北路
    818号近铁城市广场北区地下二楼N-B2-02-C商铺" },
88.      { "value": "鲜果榨汁(金沙江路和美广店)", "address": "普陀区金
    沙江路2239号金沙和美广场B1-10-6" },
89.      { "value": "开心丽果(缤谷店)", "address": "上海市长宁区威宁路
    天山路341号" },
90.      { "value": "超级鸡车(丰庄路店)", "address": "上海市嘉定区丰庄
    路240号" },
91.      { "value": "妙生活果园(北新泾店)", "address": "长宁区新渔路
    144号" },
92.      { "value": "香宜度麻辣香锅", "address": "长宁区淞虹路148号" },
93.      { "value": "凡仔汉堡(老真北路店)", "address": "上海市普陀区老
    真北路160号" },
94.      { "value": "港式小铺", "address": "上海市长宁区金钟路968号15楼
    15-105室" },
95.      { "value": "蜀香源麻辣香锅(剑河路店)", "address": "剑河路
    443-1" },
96.      { "value": "北京饺子馆", "address": "长宁区北新泾街道天山西路
    490-1号" },
97.      { "value": "饭典*新简餐(凌空SOHO店)", "address": "上海市长宁

```

```

    区金钟路968号9号楼地下一层9-83室" },
98.      { "value": "焦耳·川式快餐（金钟路店）", "address": "上海市金钟
    路633号地下一层甲部" },
99.      { "value": "动力鸡车", "address": "长宁区仙霞西路299弄3号
    101B" },
100.     { "value": "浏阳蒸菜", "address": "天山西路430号" },
101.     { "value": "四海游龙（天山西路店）", "address": "上海市长宁区天
    山西路" },
102.     { "value": "樱花食堂（凌空店）", "address": "上海市长宁区金钟路
    968号15楼15-105室" },
103.     { "value": "壹分米客家传统调制米粉（天山店）", "address": "天山西
    路428号" },
104.     { "value": "福荣祥烧腊（平溪路店）", "address": "上海市长宁区协
    和路福泉路255弄57-73号" },
105.     { "value": "速记黄焖鸡米饭", "address": "上海市长宁区北新泾街道
    金钟路180号1层01号摊位" },
106.     { "value": "红辣椒麻辣烫", "address": "上海市长宁区天山西路492
    号" },
107.     { "value": "（小杨生煎）西郊百联餐厅", "address": "长宁区仙霞西路
    88号百联2楼" },
108.     { "value": "阳阳麻辣烫", "address": "天山西路389号" },
109.     { "value": "南拳妈妈龙虾盖浇饭", "address": "普陀区金沙江路
    1699号鑫乐惠美食广场A13" }
110.   ];
111.   },
112.   handleSelect(item) {
113.     console.log(item);
114.   },
115.   handleIconClick(ev) {
116.     console.log(ev);
117.   }
118. },
119. mounted() {
120.   this.restaurants = this.loadAll();
121. }
122. }
123. </script>

```

## 远程搜索

### 从服务端搜索数据

请输入内容

三全鲜食（北新泾...  
 Hot honey 首尔炸...  
 新旺角茶餐厅  
 洸千家(天山西路店)  
 胖仙女纸杯蛋糕（...  
 贡茶  
 豪大大香鸡排超级...  
 茶芝兰（奶茶，手...

```

1. <el-autocomplete
2.   v-model="state4"
3.   :fetch-suggestions="querySearchAsync"
4.   placeholder="请输入内容"
5.   @select="handleSelect"
6. ></el-autocomplete>
7. <script>
8.   export default {
9.     data() {
10.      return {
11.        restaurants: [],
12.        state4: '',
13.        timeout: null
14.      };
15.    },
16.    methods: {
17.      loadAll() {
18.        return [
19.          { "value": "三全鲜食（北新泾店）", "address": "长宁区新渔路144

```

```

    号" },
20.      { "value": "Hot honey 首尔炸鸡（仙霞路）", "address": "上海市
    长宁区淞虹路661号" },
21.      { "value": "新旺角茶餐厅", "address": "上海市普陀区真北路988号
    创邑金沙谷6号楼113" },
22.      { "value": "泷千家(天山西路店)", "address": "天山西路438号"
    },
23.      { "value": "胖仙女纸杯蛋糕（上海凌空店）", "address": "上海市长
    宁区金钟路968号1幢18号楼一层商铺18-101" },
24.      { "value": "贡茶", "address": "上海市长宁区金钟路633号" },
25.      { "value": "豪大大香鸡排超级奶爸", "address": "上海市嘉定区曹安
    公路曹安路1685号" },
26.      { "value": "茶芝兰（奶茶，手抓饼）", "address": "上海市普陀区同
    普路1435号" },
27.      { "value": "十二泷町", "address": "上海市北翟路1444弄81号B
    幢-107" },
28.      { "value": "星移浓缩咖啡", "address": "上海市嘉定区新郁路817
    号" },
29.      { "value": "阿姨奶茶/豪大大", "address": "嘉定区曹安路1611号"
    },
30.      { "value": "新麦甜四季甜品炸鸡", "address": "嘉定区曹安公路
    2383弄55号" },
31.      { "value": "Monica摩托主题咖啡店", "address": "嘉定区江桥镇曹
    安公路2409号1F, 2383弄62号1F" },
32.      { "value": "浮生若茶（凌空soho店）", "address": "上海长宁区金
    钟路968号9号楼地下一层" },
33.      { "value": "NONO JUICE 鲜榨果汁", "address": "上海市长宁区天
    山西路119号" },
34.      { "value": "CoCo都可(北新泾店)", "address": "上海市长宁区仙霞
    西路" },
35.      { "value": "快乐柠檬（神州智慧店）", "address": "上海市长宁区天
    山西路567号1层R117号店铺" },
36.      { "value": "Merci Paul cafe", "address": "上海市普陀区光复西
    路丹巴路28弄6号楼819" },
37.      { "value": "猫山王（西郊百联店）", "address": "上海市长宁区仙霞
    西路88号第一层G05-F01-1-306" },
38.      { "value": "枪会山", "address": "上海市普陀区棕榈路" },
39.      { "value": "纵食", "address": "元丰天山花园(东门) 双流路267号"

```

```

    },
40.      { "value": "钱记", "address": "上海市长宁区天山西路" },
41.      { "value": "壹杯加", "address": "上海市长宁区通协路" },
42.      { "value": "吵哇嘀咖", "address": "上海市长宁区新泾镇金钟路999
号2幢（B幢）第01层第1-02A单元" },
43.      { "value": "爱茜茜里(西郊百联)", "address": "长宁区仙霞西路88
号1305室" },
44.      { "value": "爱茜茜里(近铁广场)", "address": "上海市普陀区真北路
818号近铁城市广场北区地下二楼N-B2-02-C商铺" },
45.      { "value": "鲜果榨汁（金沙江路和美广店）", "address": "普陀区金
沙江路2239号金沙和美广场B1-10-6" },
46.      { "value": "开心丽果(缤谷店)", "address": "上海市长宁区威宁路
天山路341号" },
47.      { "value": "超级鸡车（丰庄路店）", "address": "上海市嘉定区丰庄
路240号" },
48.      { "value": "妙生活果园（北新泾店）", "address": "长宁区新渔路
144号" },
49.      { "value": "香宜度麻辣香锅", "address": "长宁区淞虹路148号" },
50.      { "value": "凡仔汉堡（老真北路店）", "address": "上海市普陀区老
真北路160号" },
51.      { "value": "港式小铺", "address": "上海市长宁区金钟路968号15楼
15-105室" },
52.      { "value": "蜀香源麻辣香锅（剑河路店）", "address": "剑河路
443-1" },
53.      { "value": "北京饺子馆", "address": "长宁区北新泾街道天山西路
490-1号" },
54.      { "value": "饭典*新简餐（凌空SOHO店）", "address": "上海市长宁
区金钟路968号9号楼地下一层9-83室" },
55.      { "value": "焦耳·川式快餐（金钟路店）", "address": "上海市金钟
路633号地下一层甲部" },
56.      { "value": "动力鸡车", "address": "长宁区仙霞西路299弄3号
101B" },
57.      { "value": "浏阳蒸菜", "address": "天山西路430号" },
58.      { "value": "四海游龙（天山西路店）", "address": "上海市长宁区天
山西路" },
59.      { "value": "樱花食堂（凌空店）", "address": "上海市长宁区金钟路
968号15楼15-105室" },
60.      { "value": "壹分米客家传统调制米粉(天山店)", "address": "天山西

```

```

    路428号" },
61.         { "value": "福荣祥烧腊（平溪路店）", "address": "上海市长宁区协
    和路福泉路255弄57-73号" },
62.         { "value": "速记黄焖鸡米饭", "address": "上海市长宁区北新泾街道
    金钟路180号1层01号摊位" },
63.         { "value": "红辣椒麻辣烫", "address": "上海市长宁区天山西路492
    号" },
64.         { "value": "(小杨生煎)西郊百联餐厅", "address": "长宁区仙霞西路
    88号百联2楼" },
65.         { "value": "阳阳麻辣烫", "address": "天山西路389号" },
66.         { "value": "南拳妈妈龙虾盖浇饭", "address": "普陀区金沙江路
    1699号鑫乐惠美食广场A13" }
67.     ];
68. },
69.     querySearchAsync(queryString, cb) {
70.         var restaurants = this.restaurants;
71.         var results = queryString ?
    restaurants.filter(this.createStateFilter(queryString)) :
    restaurants;
72.
73.         clearTimeout(this.timeout);
74.         this.timeout = setTimeout(() => {
75.             cb(results);
76.         }, 3000 * Math.random());
77.     },
78.     createStateFilter(queryString) {
79.         return (state) => {
80.             return
    (state.value.toLowerCase().indexOf(queryString.toLowerCase()) ===
    0);
81.         };
82.     },
83.     handleSelect(item) {
84.         console.log(item);
85.     }
86. },
87.     mounted() {
88.         this.restaurants = this.loadAll();

```



```

89.     }
90.   };
91. </script>

```

## Input Attributes

参数	说明	类型	可选值	默认值
type	类型	string	text / textarea	text
value	绑定值	string / number	—	—
maxlength	最大输入长度	number	—	—
minlength	最小输入长度	number	—	—
placeholder	输入框占位文本	string	—	—
clearable	是否可清空	boolean	—	false
disabled	禁用	boolean	—	false
size	输入框尺寸，只在 type!="textarea" 时有效	string	medium / small / mini	—
prefix-icon	输入框头部图标	string	—	—
suffix-icon	输入框尾部图标	string	—	—
rows	输入框行数，只对 type="textarea" 有效	number	—	2
autosize	自适应内容高度，只对 type="textarea" 有效，可 传入对象，如，{ minRows: 2, maxRows: 6 }	boolean / object	—	false
auto- complete	原生属性，自动补全	string	on, off	off
name	原生属性	string	—	—
readonly	原生属性，是否只读	boolean	—	false
max	原生属性，设置最大值	—	—	—
min	原生属性，设置最小值	—	—	—
step	原生属性，设置输入字段的合法 数字间隔	—	—	—
resize	控制是否能被用户缩放	string	none, both, horizontal, vertical	—

autofocus	原生属性，自动获取焦点	boolean	true, false	false
form	原生属性	string	—	—
label	输入框关联的label文字	string	—	—
tabindex	输入框的tabindex	string	-	-

## Input slots

name	说明
prefix	输入框头部内容，只对 type="text" 有效
suffix	输入框尾部内容，只对 type="text" 有效
prepend	输入框前置内容，只对 type="text" 有效
append	输入框后置内容，只对 type="text" 有效

## Input Events

事件名称 说明 回调参数

blur 在 Input 失去焦点时触发 (event: Event)

focus 在 Input 获得焦点时触发 (event: Event)

change 在 Input 值改变时触发 (value: string | number)

## Input Methods

|方法名|说明|参数|

|focus|使 input 获取焦点|-|

## Autocomplete Attributes

参数	说明	类型	可选值	默认值
placeholder	输入框占位文本	string	—	—
disabled	禁用	boolean	—	false
valueKey	输入建议对象中用于显示的键名	string	—	value

value	必填值，输入绑定值	string	—	—
debounce	获取输入建议的去抖延时	number	—	300
fetch-suggestions	返回输入建议的方法，仅当你的输入建议数据 resolve 时，通过调用 callback(data:[]) 来返回它	Function(queryString, callback)	—	—
popper-class	Autocomplete 下拉列表的类名	string	—	—
trigger-on-focus	是否在输入框 focus 时显示建议列表	boolean	—	true
name	原生属性	string	—	—
select-when-unmatched	在输入没有任何匹配建议的情况下，按下回车是否触发 select 事件	boolean	—	false
label	输入框关联的label文字	string	—	—
prefix-icon	输入框头部图标	string	—	—
suffix-icon	输入框尾部图标	string	—	—

## Autocomplete slots

name	说明
prefix	输入框头部内容
suffix	输入框尾部内容
prepend	输入框前置内容
append	输入框后置内容

## Autocomplete Events

事件名称	说明	回调参数
select	点击选中建议项时触发	选中建议项

## 3.4 InputNumber 计数器

- InputNumber 计数器
  - 基础用法
  - 禁用状态
  - 步数
  - 尺寸
  - 按钮位置
  - Attributes
  - Events
  - Methods

### InputNumber 计数器

仅允许输入标准的数字值，可定义范围

#### 基础用法



要使用它，只需要在 `el-input-number` 元素中使用 `v-model` 绑定变量即可，变量的初始值即为默认值。

```
1. <template>
2.   <el-input-number v-model="num1" @change="handleChange" :min="1"
      :max="10" label="描述文字"></el-input-number>
3. </template>
4. <script>
5.   export default {
6.     data() {
7.       return {
```

```

8.         num1: 1
9.     };
10. },
11.     methods: {
12.         handleChange(value) {
13.             console.log(value);
14.         }
15.     }
16. };
17. </script>

```

## 禁用状态



`disabled` 属性接受一个 `Boolean`，设置为 `true` 即可禁用整个组件，如果你只需要控制数值在某一范围内，可以设置 `min` 属性和 `max` 属性，不设置 `min` 和 `max` 时，最小值为 0。

```

1. <template>
2.   <el-input-number v-model="num2" :disabled="true"></el-input-
   number>
3. </template>
4. <script>
5.   export default {
6.     data() {
7.       return {
8.         num2: 1
9.       }
10.    }
11.  };
12. </script>

```

## 步数

## 允许定义递增递减的步数控制



设置 `step` 属性可以控制步长，接受一个 `Number` 。

```
1. <template>
2.   <el-input-number v-model="num3" :step="2"></el-input-number>
3. </template>
4. <script>
5.   export default {
6.     data() {
7.       return {
8.         num3: 5
9.       }
10.    }
11.  };
12. </script>
```

## 尺寸

额外提供了 `medium`、`small`、`mini` 三种尺寸的数字输入框



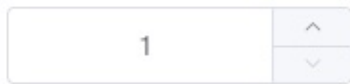
```
1. <template>
2.   <el-input-number v-model="num4"></el-input-number>
3.   <el-input-number size="medium" v-model="num5"></el-input-number>
4.   <el-input-number size="small" v-model="num6"></el-input-number>
5.   <el-input-number size="mini" v-model="num7"></el-input-number>
6. </template>
7. <script>
8.   export default {
9.     data() {
10.      return {
```

```

11.         num4: 1,
12.         num5: 1,
13.         num6: 1,
14.         num7: 1
15.     }
16. }
17. };
18. </script>

```

## 按钮位置



设置 `controls-position` 属性可以控制按钮位置。

```

1. <template>
2.   <el-input-number v-model="num8" controls-position="right"
   @change="handleChange" :min="1" :max="10"></el-input-number>
3. </template>
4. <script>
5.   export default {
6.     data() {
7.       return {
8.         num8: 1
9.       };
10.    },
11.    methods: {
12.      handleChange(value) {
13.        console.log(value);
14.      }
15.    }
16.  };
17. </script>

```

## Attributes

参数	说明	类型	可选值	默认值
value	绑定值	number	—	—
min	设置计数器允许的最小值	number	—	0
max	设置计数器允许的最大值	number	—	Infinity
step	计数器步长	number	—	1
size	计数器尺寸	string	large, small	—
disabled	是否禁用计数器	boolean	—	false
controls	是否使用控制按钮	boolean	—	true
debounce	输入时的去抖延迟，毫秒	number	—	300
controls-position	控制按钮位置	string	right	-
name	原生属性	string	—	—
label	输入框关联的label文字	string	—	—

## Events

事件名称	说明	回调参数
change	绑定值被改变时触发	最后变更的值
blur	在组件 Input 失去焦点时触发	(event: Event)
focus	在组件 Input 获得焦点时触发	(event: Event)

## Methods

方法名	说明	参数
focus	使 input 获取焦点	-



## 3.5 Select 选择器

- [Select 选择器](#)
  - [基础用法](#)
  - [有禁用选项](#)
  - [禁用状态](#)
  - [可清空单选](#)
  - [基础多选](#)
  - [自定义模板](#)
  - [分组](#)
  - [可搜索](#)
  - [远程搜索](#)
  - [创建条目](#)
  - [Select Attributes](#)
  - [Select Events](#)
  - [Option Group Attributes](#)
  - [Option Attributes](#)
  - [Methods](#)

## Select 选择器

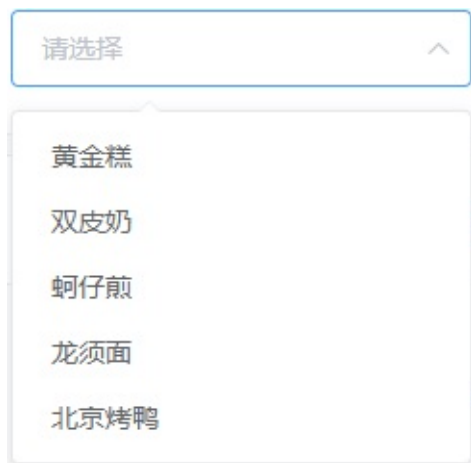
---

当选项过多时，使用下拉菜单展示并选择内容。

### 基础用法

---

适用广泛的基础单选



`v-model` 的值为当前被选中的 `el-option` 的 `value` 属性值

```
1. <template>
2.   <el-select v-model="value" placeholder="请选择">
3.     <el-option
4.       v-for="item in options"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value">
8.     </el-option>
9.   </el-select>
10. </template>
11.
12. <script>
13.   export default {
14.     data() {
15.       return {
16.         options: [{
17.           value: '选项1',
18.           label: '黄金糕'
19.         }, {
20.           value: '选项2',
21.           label: '双皮奶'
22.         }, {
23.           value: '选项3',
24.           label: '蚵仔煎'
25.         }, {
```

```

26.         value: '选项4',
27.         label: '龙须面'
28.     }, {
29.         value: '选项5',
30.         label: '北京烤鸭'
31.     }],
32.     value: ''
33. }
34. }
35. }
36. </script>

```

## 有禁用选项



在 `el-option` 中, 设定 `disabled` 值为 `true`, 即可禁用该选项

```

1. <template>
2.   <el-select v-model="value2" placeholder="请选择">
3.     <el-option
4.       v-for="item in options2"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value"
8.       :disabled="item.disabled">
9.     </el-option>
10.  </el-select>

```


```

11. </template>
12.
13. <script>
14.   export default {
15.     data() {
16.       return {
17.         options2: [{
18.           value: '选项1',
19.           label: '黄金糕'
20.         }, {
21.           value: '选项2',
22.           label: '双皮奶',
23.           disabled: true
24.         }, {
25.           value: '选项3',
26.           label: '蚵仔煎'
27.         }, {
28.           value: '选项4',
29.           label: '龙须面'
30.         }, {
31.           value: '选项5',
32.           label: '北京烤鸭'
33.         }],
34.         value2: ''
35.       }
36.     }
37.   }
38. </script>

```

## 禁用状态

### 选择器不可用状态

请选择
 

为 `el-select` 设置 `disabled` 属性，则整个选择器不可用

```
1. <template>
2.   <el-select v-model="value3" disabled placeholder="请选择">
3.     <el-option
4.       v-for="item in options"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value">
8.     </el-option>
9.   </el-select>
10. </template>
11.
12. <script>
13.   export default {
14.     data() {
15.       return {
16.         options: [{
17.           value: '选项1',
18.           label: '黄金糕'
19.         }, {
20.           value: '选项2',
21.           label: '双皮奶'
22.         }, {
23.           value: '选项3',
24.           label: '蚵仔煎'
25.         }, {
26.           value: '选项4',
27.           label: '龙须面'
28.         }, {
29.           value: '选项5',
30.           label: '北京烤鸭'
31.         }
32.       ],
33.       value3: ''
34.     }
35.   }
36. </script>
```

## 可清空单选

包含清空按钮，可将选择器清空为初始状态



为 `el-select` 设置 `clearable` 属性，则可将选择器清空。需要注意的是，`clearable` 属性仅适用于单选。

```
1. <template>
2.   <el-select v-model="value4" clearable placeholder="请选择">
3.     <el-option
4.       v-for="item in options"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value">
8.     </el-option>
9.   </el-select>
10. </template>
11.
12. <script>
13.   export default {
14.     data() {
15.       return {
16.         options: [{
17.           value: '选项1',
18.           label: '黄金糕'
19.         }, {
20.           value: '选项2',
21.           label: '双皮奶'
22.         }, {
23.           value: '选项3',
24.           label: '蚵仔煎'
25.         }, {
26.           value: '选项4',
27.           label: '龙须面'
28.         }, {
```

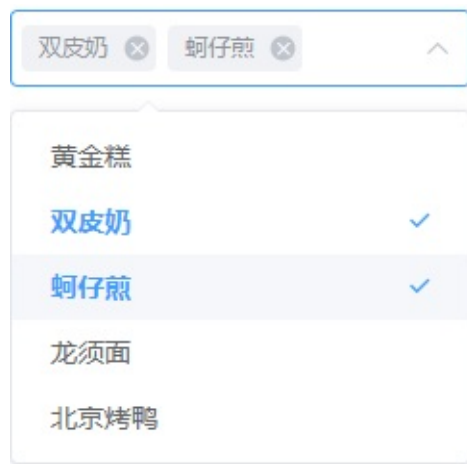
```

29.         value: '选项5',
30.         label: '北京烤鸭'
31.     }],
32.     value4: ''
33. }
34. }
35. }
36. </script>

```

## 基础多选

适用性较广的基础多选，用 Tag 展示已选项



为 `el-select` 设置 `multiple` 属性即可启用多选，此时 `v-model` 的值为当前选中值所组成的数组。默认情况下选中值会以 Tag 的形式展现，你也可以设置 `collapse-tags` 属性将它们合并为一段文字。

```

1. <template>
2.   <el-select v-model="value5" multiple placeholder="请选择">
3.     <el-option
4.       v-for="item in options"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value">
8.     </el-option>
9.   </el-select>

```

```
10.
11.   <el-select
12.     v-model="value11"
13.     multiple
14.     collapse-tags
15.     style="margin-left: 20px;"
16.     placeholder="请选择">
17.     <el-option
18.       v-for="item in options"
19.       :key="item.value"
20.       :label="item.label"
21.       :value="item.value">
22.     </el-option>
23.   </el-select>
24. </template>
25.
26. <script>
27.   export default {
28.     data() {
29.       return {
30.         options: [{
31.           value: '选项1',
32.           label: '黄金糕'
33.         }, {
34.           value: '选项2',
35.           label: '双皮奶'
36.         }, {
37.           value: '选项3',
38.           label: '蚵仔煎'
39.         }, {
40.           value: '选项4',
41.           label: '龙须面'
42.         }, {
43.           value: '选项5',
44.           label: '北京烤鸭'
45.         }
46.       ],
47.       value5: [],
48.       value11: []
```



```

48.     }
49.     }
50.   }
51. </script>

```

## 自定义模板

可以自定义备选项



将自定义的 HTML 模板插入 `el-option` 的 `slot` 中即可。

```

1. <template>
2.   <el-select v-model="value6" placeholder="请选择">
3.     <el-option
4.       v-for="item in cities"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value">
8.       <span style="float: left">{{ item.label }}</span>
9.       <span style="float: right; color: #8492a6; font-size: 13px">
10.        {{ item.value }}</span>
11.     </el-option>
12.   </el-select>
13. </template>

```

```
14. <script>
15.   export default {
16.     data() {
17.       return {
18.         cities: [{
19.           value: 'Beijing',
20.           label: '北京'
21.         }, {
22.           value: 'Shanghai',
23.           label: '上海'
24.         }, {
25.           value: 'Nanjing',
26.           label: '南京'
27.         }, {
28.           value: 'Chengdu',
29.           label: '成都'
30.         }, {
31.           value: 'Shenzhen',
32.           label: '深圳'
33.         }, {
34.           value: 'Guangzhou',
35.           label: '广州'
36.         }
37.       ],
38.       value6: ''
39.     }
40.   }
41. </script>
```

## 分组

### 备选项进行分组展示



使用 `el-option-group` 对备选项进行分组，它的 `label` 属性为分组名

```

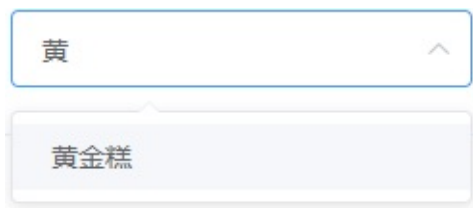
1. <template>
2.   <el-select v-model="value7" placeholder="请选择">
3.     <el-option-group
4.       v-for="group in options3"
5.       :key="group.label"
6.       :label="group.label">
7.       <el-option
8.         v-for="item in group.options"
9.         :key="item.value"
10.        :label="item.label"
11.        :value="item.value">
12.       </el-option>
13.     </el-option-group>
14.   </el-select>
15. </template>
16.
17. <script>
18.   export default {
19.     data() {
20.       return {
21.         options3: [{
22.           label: '热门城市',

```

```
23.         options: [{
24.             value: 'Shanghai',
25.             label: '上海'
26.         }, {
27.             value: 'Beijing',
28.             label: '北京'
29.         }]
30.     }, {
31.         label: '城市名',
32.         options: [{
33.             value: 'Chengdu',
34.             label: '成都'
35.         }, {
36.             value: 'Shenzhen',
37.             label: '深圳'
38.         }, {
39.             value: 'Guangzhou',
40.             label: '广州'
41.         }, {
42.             value: 'Dalian',
43.             label: '大连'
44.         }]
45.     }],
46.     value7: ''
47. }
48. }
49. }
50. </script>
```

## 可搜索

可以利用搜索功能快速查找选项



为 `el-select` 添加 `filterable` 属性即可启用搜索功能。默认情况下，`Select` 会找出所有 `label` 属性包含输入值的选项。如果希望使用其他的搜索逻辑，可以通过传入一个 `filter-method` 来实现。`filter-method` 为一个 `Function`，它会在输入值发生变化时调用，参数为当前输入值。

```

1. <template>
2.   <el-select v-model="value8" filterable placeholder="请选择">
3.     <el-option
4.       v-for="item in options"
5.       :key="item.value"
6.       :label="item.label"
7.       :value="item.value">
8.     </el-option>
9.   </el-select>
10. </template>
11.
12. <script>
13.   export default {
14.     data() {
15.       return {
16.         options: [{
17.           value: '选项1',
18.           label: '黄金糕'
19.         }, {
20.           value: '选项2',
21.           label: '双皮奶'
22.         }, {
23.           value: '选项3',
24.           label: '蚵仔煎'
25.         }, {

```

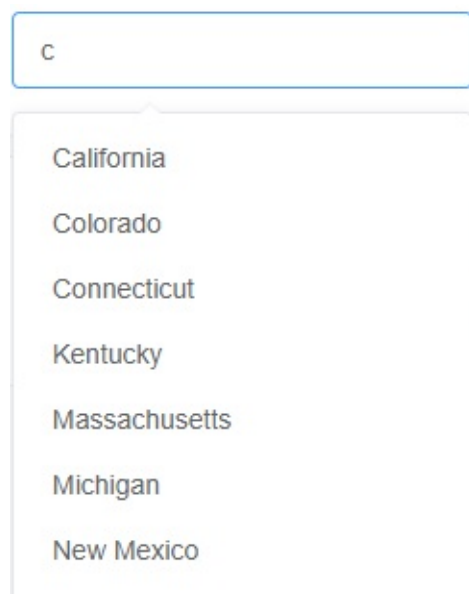
```

26.         value: '选项4',
27.         label: '龙须面'
28.     }, {
29.         value: '选项5',
30.         label: '北京烤鸭'
31.     }],
32.     value8: ''
33. }
34. }
35. }
36. </script>

```

## 远程搜索

从服务器搜索数据，输入关键字进行查找



为了启用远程搜索，需要将 `filterable` 和 `remote` 设置为 `true`，同时传入一个 `remote-method`。`remote-method` 为一个 `Function`，它会在输入值发生变化时调用，参数为当前输入值。需要注意的是，如果 `el-option` 是通过 `v-for` 指令渲染出来的，此时需要为 `el-option` 添加 `key` 属性，且其值需具有唯一性，比如此例中的 `item.value`。

```
1. <template>
2.   <el-select
3.     v-model="value9"
4.     multiple
5.     filterable
6.     remote
7.     reserve-keyword
8.     placeholder="请输入关键词"
9.     :remote-method="remoteMethod"
10.    :loading="loading">
11.    <el-option
12.      v-for="item in options4"
13.      :key="item.value"
14.      :label="item.label"
15.      :value="item.value">
16.    </el-option>
17.  </el-select>
18. </template>
19.
20. <script>
21.   export default {
22.     data() {
23.       return {
24.         options4: [],
25.         value9: [],
26.         list: [],
27.         loading: false,
28.         states: ["Alabama", "Alaska", "Arizona",
29.                  "Arkansas", "California", "Colorado",
30.                  "Connecticut", "Delaware", "Florida",
31.                  "Georgia", "Hawaii", "Idaho", "Illinois",
32.                  "Indiana", "Iowa", "Kansas", "Kentucky",
33.                  "Louisiana", "Maine", "Maryland",
34.                  "Massachusetts", "Michigan", "Minnesota",
35.                  "Mississippi", "Missouri", "Montana",
36.                  "Nebraska", "Nevada", "New Hampshire",
37.                  "New Jersey", "New Mexico", "New York",
38.                  "North Carolina", "North Dakota", "Ohio",
```

```

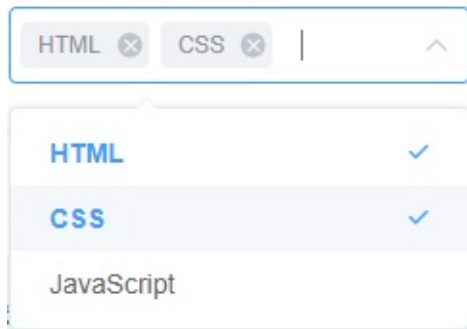
39.         "Oklahoma", "Oregon", "Pennsylvania",
40.         "Rhode Island", "South Carolina",
41.         "South Dakota", "Tennessee", "Texas",
42.         "Utah", "Vermont", "Virginia",
43.         "Washington", "West Virginia", "Wisconsin",
44.         "Wyoming"]
45.     }
46. },
47.     mounted() {
48.         this.list = this.states.map(item => {
49.             return { value: item, label: item };
50.         });
51.     },
52.     methods: {
53.         remoteMethod(query) {
54.             if (query !== '') {
55.                 this.loading = true;
56.                 setTimeout(() => {
57.                     this.loading = false;
58.                     this.options4 = this.list.filter(item => {
59.                         return item.label.toLowerCase()
60.                             .indexOf(query.toLowerCase()) > -1;
61.                     });
62.                 }, 200);
63.             } else {
64.                 this.options4 = [];
65.             }
66.         }
67.     }
68. }
69. </script>

```

## 创建条目

可以创建并选中选项中不存在的条目





使用 `allow-create` 属性即可通过在输入框中输入文字来创建新的条目。注意此时 `filterable` 必须为真。本例还使用了 `default-first-option` 属性，在该属性打开的情况下，按下回车就可以选中当前选项列表中的第一个选项，无需使用鼠标或键盘方向键进行定位。

```

1. <template>
2.   <el-select
3.     v-model="value10"
4.     multiple
5.     filterable
6.     allow-create
7.     default-first-option
8.     placeholder="请选择文章标签">
9.     <el-option
10.      v-for="item in options5"
11.      :key="item.value"
12.      :label="item.label"
13.      :value="item.value">
14.     </el-option>
15.   </el-select>
16. </template>
17.
18. <script>
19.   export default {
20.     data() {
21.       return {
22.         options5: [{
23.           value: 'HTML',
24.           label: 'HTML'

```

```

25.         }, {
26.             value: 'CSS',
27.             label: 'CSS'
28.         }, {
29.             value: 'JavaScript',
30.             label: 'JavaScript'
31.         }],
32.         value10: []
33.     }
34. }
35. }
36. </script>

```

如果 `Select` 的绑定值为对象类型，请务必指定 `value-key` 作为它的唯一性标识。

## Select Attributes

参数	说明	类型	可选值	默认值
<code>multiple</code>	是否多选	boolean	—	false
<code>disabled</code>	是否禁用	boolean	—	false
<code>value-key</code>	作为 <code>value</code> 唯一标识的键名，绑定值为对象类型时必填	string	—	value
<code>size</code>	输入框尺寸	string	large/small/mini	—
<code>clearable</code>	单选时是否可以清空选项	boolean	—	false
<code>collapse-tags</code>	多选时是否将选中值按文字的形式展示	boolean	—	false
<code>multiple-limit</code>	多选时用户最多可以选择的项目数，为 0 则不限制	number	—	0
<code>name</code>	select input 的 name 属性	string	—	—
<code>placeholder</code>	占位符	string	—	请选择
<code>filterable</code>	是否可搜索	boolean	—	false
<code>allow-create</code>	是否允许用户创建新条目，需配合 <code>filterable</code> 使用	boolean	—	false
<code>filter-method</code>	自定义搜索方法	function	—	—

remote	是否为远程搜索	boolean	—	false
remote-method	远程搜索方法	function	—	—
loading	是否正在从远程获取数据	boolean	—	false
loading-text	远程加载时显示的文字	string	—	加载中
no-match-text	搜索条件无匹配时显示的文字	string	—	无匹配数据
no-data-text	选项为空时显示的文字	string	—	无数据
popper-class	Select 下拉框的类名	string	—	—
reserve-keyword	多选且可搜索时，是否在选中一个选项后保留当前的搜索关键词	boolean	—	false
default-first-option	在输入框按下回车，选择第一个匹配项。需配合 filterable 或 remote 使用	boolean	-	false

## Select Events

事件名称	说明	回调参数
change	选中值发生变化时触发	目前的选中值
visible-change	下拉框出现/隐藏时触发	出现则为 true，隐藏则为 false
remove-tag	多选模式下移除tag时触发	移除的tag值
clear	可清空的单选模式下用户点击清空按钮时触发	—
blur	当 input 失去焦点时触发	(event: Event)
focus	当 input 获得焦点时触发	(event: Event)

## Option Group Attributes

参数	说明	类型	可选值	默认值
label	分组的组名	string	—	—
disabled	是否将该分组下所有选项置为禁用	boolean	—	false

## Option Attributes

参数	说明	类型	可选值	默认值
value	选项的值	string/number/object	—	—
label	选项的标签，若不设置则默认与 value 相同	string/number	—	—
disabled	是否禁用该选项	boolean	—	false

## Methods

方法名	说明	参数
focus	使 input 获取焦点	-

## 3.6 Cascader 级联选择器

- [Cascader 级联选择器](#)
  - [基础用法](#)
  - [禁用选项](#)
  - [仅显示最后一级](#)
  - [默认值](#)
  - [选择即改变](#)
  - [动态加载次级选项](#)
  - [可搜索](#)
  - [Attributes](#)
  - [props](#)
  - [Events](#)

### Cascader 级联选择器

---

当一个数据集合有清晰的层级结构时，可通过级联选择器逐级查看并选择。

#### 基础用法

---

有两种触发子菜单的方式



只需为 Cascader 的 `options` 属性指定选项数组即可渲染出一个级联选择器。通过 `expand-trigger` 可以定义展开子级菜单的触发方式。本例还展示了 `change` 事件，它的参数为 Cascader 的绑定值：一个由各级菜单的值所组成的数组。

```

1. <div class="block">
2.   <span class="demonstration">默认 click 触发子菜单</span>
3.   <el-cascader
4.     :options="options"
5.     v-model="selectedOptions"
6.     @change="handleChange">
7.   </el-cascader>
8. </div>
9. <div class="block">
10.  <span class="demonstration">hover 触发子菜单</span>
11.  <el-cascader
12.    expand-trigger="hover"
13.    :options="options"
14.    v-model="selectedOptions2"
15.    @change="handleChange">
16.  </el-cascader>
17. </div>
18.
19. <script>

```

```
20.   export default {
21.     data() {
22.       return {
23.         options: [{
24.           value: 'zhinan',
25.           label: '指南',
26.           children: [{
27.             value: 'shejiyuanze',
28.             label: '设计原则',
29.             children: [{
30.               value: 'yizhi',
31.               label: '一致'
32.             }, {
33.               value: 'fankui',
34.               label: '反馈'
35.             }, {
36.               value: 'xiaolv',
37.               label: '效率'
38.             }, {
39.               value: 'kekong',
40.               label: '可控'
41.             }]
42.           }, {
43.             value: 'daohang',
44.             label: '导航',
45.             children: [{
46.               value: 'cexiangdaohang',
47.               label: '侧向导航'
48.             }, {
49.               value: 'dingbudaohang',
50.               label: '顶部导航'
51.             }]
52.           }]
53.       }, {
54.         value: 'zujian',
55.         label: '组件',
56.         children: [{
57.           value: 'basic',
```

```

58.         label: 'Basic',
59.         children: [{
60.             value: 'layout',
61.             label: 'Layout 布局'
62.         }, {
63.             value: 'color',
64.             label: 'Color 色彩'
65.         }, {
66.             value: 'typography',
67.             label: 'Typography 字体'
68.         }, {
69.             value: 'icon',
70.             label: 'Icon 图标'
71.         }, {
72.             value: 'button',
73.             label: 'Button 按钮'
74.         }]
75.     }, {
76.         value: 'form',
77.         label: 'Form',
78.         children: [{
79.             value: 'radio',
80.             label: 'Radio 单选框'
81.         }, {
82.             value: 'checkbox',
83.             label: 'Checkbox 多选框'
84.         }, {
85.             value: 'input',
86.             label: 'Input 输入框'
87.         }, {
88.             value: 'input-number',
89.             label: 'InputNumber 计数器'
90.         }, {
91.             value: 'select',
92.             label: 'Select 选择器'
93.         }, {
94.             value: 'cascader',
95.             label: 'Cascader 级联选择器'

```



```
96.         }, {
97.             value: 'switch',
98.             label: 'Switch 开关'
99.         }, {
100.            value: 'slider',
101.            label: 'Slider 滑块'
102.        }, {
103.            value: 'time-picker',
104.            label: 'TimePicker 时间选择器'
105.        }, {
106.            value: 'date-picker',
107.            label: 'DatePicker 日期选择器'
108.        }, {
109.            value: 'datetime-picker',
110.            label: 'DateTimePicker 日期时间选择器'
111.        }, {
112.            value: 'upload',
113.            label: 'Upload 上传'
114.        }, {
115.            value: 'rate',
116.            label: 'Rate 评分'
117.        }, {
118.            value: 'form',
119.            label: 'Form 表单'
120.        }
121.    ], {
122.        value: 'data',
123.        label: 'Data',
124.        children: [{
125.            value: 'table',
126.            label: 'Table 表格'
127.        }, {
128.            value: 'tag',
129.            label: 'Tag 标签'
130.        }, {
131.            value: 'progress',
132.            label: 'Progress 进度条'
133.        }, {
```

```
134.         value: 'tree',
135.         label: 'Tree 树形控件'
136.     }, {
137.         value: 'pagination',
138.         label: 'Pagination 分页'
139.     }, {
140.         value: 'badge',
141.         label: 'Badge 标记'
142.     }]
143. }, {
144.     value: 'notice',
145.     label: 'Notice',
146.     children: [{
147.         value: 'alert',
148.         label: 'Alert 警告'
149.     }, {
150.         value: 'loading',
151.         label: 'Loading 加载'
152.     }, {
153.         value: 'message',
154.         label: 'Message 消息提示'
155.     }, {
156.         value: 'message-box',
157.         label: 'MessageBox 弹框'
158.     }, {
159.         value: 'notification',
160.         label: 'Notification 通知'
161.     }]
162. }, {
163.     value: 'navigation',
164.     label: 'Navigation',
165.     children: [{
166.         value: 'menu',
167.         label: 'NavMenu 导航菜单'
168.     }, {
169.         value: 'tabs',
170.         label: 'Tabs 标签页'
171.     }, {
```

```
172.         value: 'breadcrumb',
173.         label: 'Breadcrumb 面包屑'
174.     }, {
175.         value: 'dropdown',
176.         label: 'Dropdown 下拉菜单'
177.     }, {
178.         value: 'steps',
179.         label: 'Steps 步骤条'
180.     }]
181. }, {
182.     value: 'others',
183.     label: 'Others',
184.     children: [{
185.         value: 'dialog',
186.         label: 'Dialog 对话框'
187.     }, {
188.         value: 'tooltip',
189.         label: 'Tooltip 文字提示'
190.     }, {
191.         value: 'popover',
192.         label: 'Popover 弹出框'
193.     }, {
194.         value: 'card',
195.         label: 'Card 卡片'
196.     }, {
197.         value: 'carousel',
198.         label: 'Carousel 走马灯'
199.     }, {
200.         value: 'collapse',
201.         label: 'Collapse 折叠面板'
202.     }]
203. }]
204. }, {
205.     value: 'ziyuan',
206.     label: '资源',
207.     children: [{
208.         value: 'axure',
209.         label: 'Axure Components'
```

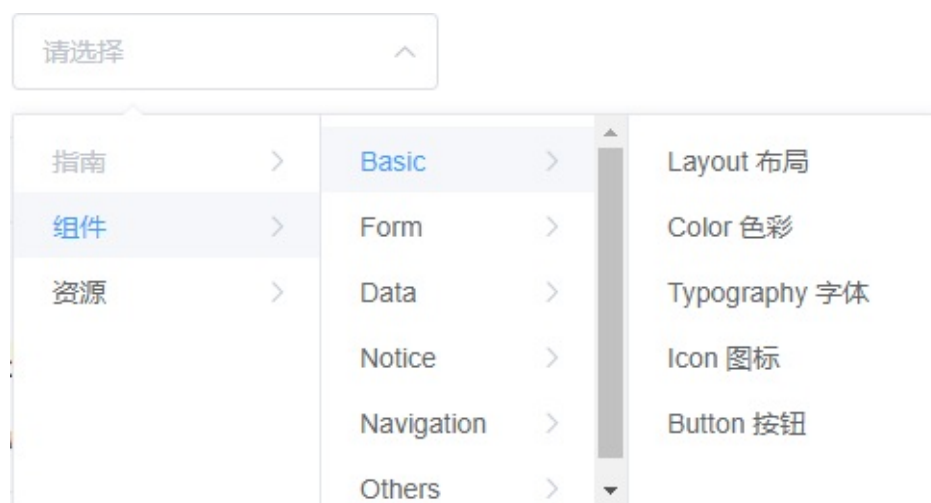
```

210.         }, {
211.             value: 'sketch',
212.             label: 'Sketch Templates'
213.         }, {
214.             value: 'jiaohu',
215.             label: '组件交互文档'
216.         }]
217.     ]],
218.     selectedOptions: [],
219.     selectedOptions2: []
220. };
221. },
222. methods: {
223.     handleChange(value) {
224.         console.log(value);
225.     }
226. }
227. };
228. </script>

```

## 禁用选项

通过在数据源中设置 `disabled` 字段来声明该选项是禁用的



本例中，`options` 指定的数组中的第一个元素含有 `disabled: true` 键值对，因此是禁用的。在默认情况下，Cascader 会检查数据中每一

项的 `disabled` 字段是否为 `true`，如果你的数据中表示禁用含义的字段名不为 `disabled`，可以通过 `props` 属性来指定（详见下方 API 表格）。当然，`value`、`label` 和 `children` 这三个字段名也可以通过同样的方式指定。

```

1. <el-cascader
2.   :options="optionsWithDisabled"
3. ></el-cascader>
4. <script>
5.   export default {
6.     data() {
7.       return {
8.         optionsWithDisabled: [{
9.           value: 'zhinan',
10.          label: '指南',
11.          disabled: true,
12.          children: [{
13.            value: 'shejiyuanze',
14.            label: '设计原则',
15.            children: [{
16.              value: 'yizhi',
17.              label: '一致'
18.            }, {
19.              value: 'fankui',
20.              label: '反馈'
21.            }, {
22.              value: 'xiaolv',
23.              label: '效率'
24.            }, {
25.              value: 'kekong',
26.              label: '可控'
27.            }
28.          ], {
29.            value: 'daohang',
30.            label: '导航',
31.            children: [{
32.              value: 'cexiangdaohang',

```

```

33.         label: '侧向导航'
34.     }, {
35.         value: 'dingbudaohang',
36.         label: '顶部导航'
37.     }]
38. ]
39. }, {
40.     value: 'zujian',
41.     label: '组件',
42.     children: [{
43.         value: 'basic',
44.         label: 'Basic',
45.         children: [{
46.             value: 'layout',
47.             label: 'Layout 布局'
48.         }, {
49.             value: 'color',
50.             label: 'Color 色彩'
51.         }, {
52.             value: 'typography',
53.             label: 'Typography 字体'
54.         }, {
55.             value: 'icon',
56.             label: 'Icon 图标'
57.         }, {
58.             value: 'button',
59.             label: 'Button 按钮'
60.         }]
61.     }, {
62.         value: 'form',
63.         label: 'Form',
64.         children: [{
65.             value: 'radio',
66.             label: 'Radio 单选框'
67.         }, {
68.             value: 'checkbox',
69.             label: 'Checkbox 多选框'
70.         }, {

```

```
71.         value: 'input',
72.         label: 'Input 输入框'
73.     }, {
74.         value: 'input-number',
75.         label: 'InputNumber 计数器'
76.     }, {
77.         value: 'select',
78.         label: 'Select 选择器'
79.     }, {
80.         value: 'cascader',
81.         label: 'Cascader 级联选择器'
82.     }, {
83.         value: 'switch',
84.         label: 'Switch 开关'
85.     }, {
86.         value: 'slider',
87.         label: 'Slider 滑块'
88.     }, {
89.         value: 'time-picker',
90.         label: 'TimePicker 时间选择器'
91.     }, {
92.         value: 'date-picker',
93.         label: 'DatePicker 日期选择器'
94.     }, {
95.         value: 'datetime-picker',
96.         label: 'DateTimePicker 日期时间选择器'
97.     }, {
98.         value: 'upload',
99.         label: 'Upload 上传'
100.    }, {
101.        value: 'rate',
102.        label: 'Rate 评分'
103.    }, {
104.        value: 'form',
105.        label: 'Form 表单'
106.    }]
107.    }, {
108.        value: 'data',
```

```
109.         label: 'Data',
110.         children: [{
111.             value: 'table',
112.             label: 'Table 表格'
113.         }, {
114.             value: 'tag',
115.             label: 'Tag 标签'
116.         }, {
117.             value: 'progress',
118.             label: 'Progress 进度条'
119.         }, {
120.             value: 'tree',
121.             label: 'Tree 树形控件'
122.         }, {
123.             value: 'pagination',
124.             label: 'Pagination 分页'
125.         }, {
126.             value: 'badge',
127.             label: 'Badge 标记'
128.         }
129.     ], {
130.         value: 'notice',
131.         label: 'Notice',
132.         children: [{
133.             value: 'alert',
134.             label: 'Alert 警告'
135.         }, {
136.             value: 'loading',
137.             label: 'Loading 加载'
138.         }, {
139.             value: 'message',
140.             label: 'Message 消息提示'
141.         }, {
142.             value: 'message-box',
143.             label: 'MessageBox 弹框'
144.         }, {
145.             value: 'notification',
146.             label: 'Notification 通知'
```



```
147.         ]]  
148.     }, {  
149.         value: 'navigation',  
150.         label: 'Navigation',  
151.         children: [{  
152.             value: 'menu',  
153.             label: 'NavMenu 导航菜单'  
154.         }, {  
155.             value: 'tabs',  
156.             label: 'Tabs 标签页'  
157.         }, {  
158.             value: 'breadcrumb',  
159.             label: 'Breadcrumb 面包屑'  
160.         }, {  
161.             value: 'dropdown',  
162.             label: 'Dropdown 下拉菜单'  
163.         }, {  
164.             value: 'steps',  
165.             label: 'Steps 步骤条'  
166.         }]  
167.     }, {  
168.         value: 'others',  
169.         label: 'Others',  
170.         children: [{  
171.             value: 'dialog',  
172.             label: 'Dialog 对话框'  
173.         }, {  
174.             value: 'tooltip',  
175.             label: 'Tooltip 文字提示'  
176.         }, {  
177.             value: 'popover',  
178.             label: 'Popover 弹出框'  
179.         }, {  
180.             value: 'card',  
181.             label: 'Card 卡片'  
182.         }, {  
183.             value: 'carousel',  
184.             label: 'Carousel 走马灯'
```

```
185.         }, {
186.             value: 'collapse',
187.             label: 'Collapse 折叠面板'
188.         }]
189.     }]
190. }, {
191.     value: 'ziyuan',
192.     label: '资源',
193.     children: [{
194.         value: 'axure',
195.         label: 'Axure Components'
196.     }, {
197.         value: 'sketch',
198.         label: 'Sketch Templates'
199.     }, {
200.         value: 'jiaohu',
201.         label: '组件交互文档'
202.     }]
203. }]
204. };
205. }
206. };
207. </script>
```

## 仅显示最后一级

可以仅在输入框中显示选中项最后一级的标签，而不是选中项所在的完整路径。



属性 `show-all-levels` 定义了是否显示完整的路径，将其赋值为`false` 则仅显示最后一级

```

1. <el-cascader
2.   :options="options"
3.   :show-all-levels="false"
4. ></el-cascader>
5. <script>
6.   export default {
7.     data() {
8.       return {
9.         options: [{
10.          value: 'zhinan',
11.          label: '指南',
12.          children: [{
13.            value: 'shejiyuanze',
14.            label: '设计原则',
15.            children: [{
16.              value: 'yizhi',
17.              label: '一致'
18.            }, {
19.              value: 'fankui',
20.              label: '反馈'
21.            }, {
22.              value: 'xiaolv',
23.              label: '效率'

```

```

24.         }, {
25.             value: 'kekong',
26.             label: '可控'
27.         }]
28.     }, {
29.         value: 'daohang',
30.         label: '导航',
31.         children: [{
32.             value: 'cexiangdaohang',
33.             label: '侧向导航'
34.         }, {
35.             value: 'dingbudaohang',
36.             label: '顶部导航'
37.         }]
38.     }]
39. }, {
40.     value: 'zujian',
41.     label: '组件',
42.     children: [{
43.         value: 'basic',
44.         label: 'Basic',
45.         children: [{
46.             value: 'layout',
47.             label: 'Layout 布局'
48.         }, {
49.             value: 'color',
50.             label: 'Color 色彩'
51.         }, {
52.             value: 'typography',
53.             label: 'Typography 字体'
54.         }, {
55.             value: 'icon',
56.             label: 'Icon 图标'
57.         }, {
58.             value: 'button',
59.             label: 'Button 按钮'
60.         }]
61.     }, {

```

```
62.         value: 'form',
63.         label: 'Form',
64.         children: [{
65.             value: 'radio',
66.             label: 'Radio 单选框'
67.         }, {
68.             value: 'checkbox',
69.             label: 'Checkbox 多选框'
70.         }, {
71.             value: 'input',
72.             label: 'Input 输入框'
73.         }, {
74.             value: 'input-number',
75.             label: 'InputNumber 计数器'
76.         }, {
77.             value: 'select',
78.             label: 'Select 选择器'
79.         }, {
80.             value: 'cascader',
81.             label: 'Cascader 级联选择器'
82.         }, {
83.             value: 'switch',
84.             label: 'Switch 开关'
85.         }, {
86.             value: 'slider',
87.             label: 'Slider 滑块'
88.         }, {
89.             value: 'time-picker',
90.             label: 'TimePicker 时间选择器'
91.         }, {
92.             value: 'date-picker',
93.             label: 'DatePicker 日期选择器'
94.         }, {
95.             value: 'datetime-picker',
96.             label: 'DateTimePicker 日期时间选择器'
97.         }, {
98.             value: 'upload',
99.             label: 'Upload 上传'
```

```
100.         }, {
101.             value: 'rate',
102.             label: 'Rate 评分'
103.         }, {
104.             value: 'form',
105.             label: 'Form 表单'
106.         }]
107.     }, {
108.         value: 'data',
109.         label: 'Data',
110.         children: [{
111.             value: 'table',
112.             label: 'Table 表格'
113.         }, {
114.             value: 'tag',
115.             label: 'Tag 标签'
116.         }, {
117.             value: 'progress',
118.             label: 'Progress 进度条'
119.         }, {
120.             value: 'tree',
121.             label: 'Tree 树形控件'
122.         }, {
123.             value: 'pagination',
124.             label: 'Pagination 分页'
125.         }, {
126.             value: 'badge',
127.             label: 'Badge 标记'
128.         }]
129.     }, {
130.         value: 'notice',
131.         label: 'Notice',
132.         children: [{
133.             value: 'alert',
134.             label: 'Alert 警告'
135.         }, {
136.             value: 'loading',
137.             label: 'Loading 加载'
```

```
138.         }, {
139.             value: 'message',
140.             label: 'Message 消息提示'
141.         }, {
142.             value: 'message-box',
143.             label: 'MessageBox 弹框'
144.         }, {
145.             value: 'notification',
146.             label: 'Notification 通知'
147.         }]
148.     }, {
149.         value: 'navigation',
150.         label: 'Navigation',
151.         children: [{
152.             value: 'menu',
153.             label: 'NavMenu 导航菜单'
154.         }, {
155.             value: 'tabs',
156.             label: 'Tabs 标签页'
157.         }, {
158.             value: 'breadcrumb',
159.             label: 'Breadcrumb 面包屑'
160.         }, {
161.             value: 'dropdown',
162.             label: 'Dropdown 下拉菜单'
163.         }, {
164.             value: 'steps',
165.             label: 'Steps 步骤条'
166.         }]
167.     }, {
168.         value: 'others',
169.         label: 'Others',
170.         children: [{
171.             value: 'dialog',
172.             label: 'Dialog 对话框'
173.         }, {
174.             value: 'tooltip',
175.             label: 'Tooltip 文字提示'
```

```

176.         }, {
177.             value: 'popover',
178.             label: 'Popover 弹出框'
179.         }, {
180.             value: 'card',
181.             label: 'Card 卡片'
182.         }, {
183.             value: 'carousel',
184.             label: 'Carousel 走马灯'
185.         }, {
186.             value: 'collapse',
187.             label: 'Collapse 折叠面板'
188.         }]
189.     ]
190. }, {
191.     value: 'ziyuan',
192.     label: '资源',
193.     children: [{
194.         value: 'axure',
195.         label: 'Axure Components'
196.     }, {
197.         value: 'sketch',
198.         label: 'Sketch Templates'
199.     }, {
200.         value: 'jiaohu',
201.         label: '组件交互文档'
202.     }]
203. }
204. ];
205. }
206. ];
207. </script>

```

## 默认值

组件 / Data / Tag 标签



默认值通过数组的方式指定。

```
1. <el-cascader
2.   :options="options"
3.   v-model="selectedOptions3"
4. ></el-cascader>
5. <script>
6.   export default {
7.     data() {
8.       return {
9.         options: [{
10.          value: 'zhinan',
11.          label: '指南',
12.          children: [{
13.            value: 'shejiyuanze',
14.            label: '设计原则',
15.            children: [{
16.              value: 'yizhi',
17.              label: '一致'
18.            }, {
19.              value: 'fankui',
20.              label: '反馈'
21.            }, {
22.              value: 'xiaolv',
23.              label: '效率'
24.            }, {
25.              value: 'kekong',
26.              label: '可控'
27.            }
28.          ], {
29.            value: 'daohang',
30.            label: '导航',
31.            children: [{
32.              value: 'cexiangdaohang',
33.              label: '侧向导航'
34.            }, {
35.              value: 'dingbudaohang',
36.              label: '顶部导航'
```

```
37.         }]  
38.     }]  
39.     }, {  
40.         value: 'zujian',  
41.         label: '组件',  
42.         children: [{  
43.             value: 'basic',  
44.             label: 'Basic',  
45.             children: [{  
46.                 value: 'layout',  
47.                 label: 'Layout 布局'  
48.             }, {  
49.                 value: 'color',  
50.                 label: 'Color 色彩'  
51.             }, {  
52.                 value: 'typography',  
53.                 label: 'Typography 字体'  
54.             }, {  
55.                 value: 'icon',  
56.                 label: 'Icon 图标'  
57.             }, {  
58.                 value: 'button',  
59.                 label: 'Button 按钮'  
60.             }]  
61.         }, {  
62.             value: 'form',  
63.             label: 'Form',  
64.             children: [{  
65.                 value: 'radio',  
66.                 label: 'Radio 单选框'  
67.             }, {  
68.                 value: 'checkbox',  
69.                 label: 'Checkbox 多选框'  
70.             }, {  
71.                 value: 'input',  
72.                 label: 'Input 输入框'  
73.             }, {  
74.                 value: 'input-number',
```

```

75.         label: 'InputNumber 计数器'
76.     }, {
77.         value: 'select',
78.         label: 'Select 选择器'
79.     }, {
80.         value: 'cascader',
81.         label: 'Cascader 级联选择器'
82.     }, {
83.         value: 'switch',
84.         label: 'Switch 开关'
85.     }, {
86.         value: 'slider',
87.         label: 'Slider 滑块'
88.     }, {
89.         value: 'time-picker',
90.         label: 'TimePicker 时间选择器'
91.     }, {
92.         value: 'date-picker',
93.         label: 'DatePicker 日期选择器'
94.     }, {
95.         value: 'datetime-picker',
96.         label: 'DateTimePicker 日期时间选择器'
97.     }, {
98.         value: 'upload',
99.         label: 'Upload 上传'
100.    }, {
101.        value: 'rate',
102.        label: 'Rate 评分'
103.    }, {
104.        value: 'form',
105.        label: 'Form 表单'
106.    }]
107. }, {
108.     value: 'data',
109.     label: 'Data',
110.     children: [{
111.         value: 'table',
112.         label: 'Table 表格'

```

```
113.         }, {
114.             value: 'tag',
115.             label: 'Tag 标签'
116.         }, {
117.             value: 'progress',
118.             label: 'Progress 进度条'
119.         }, {
120.             value: 'tree',
121.             label: 'Tree 树形控件'
122.         }, {
123.             value: 'pagination',
124.             label: 'Pagination 分页'
125.         }, {
126.             value: 'badge',
127.             label: 'Badge 标记'
128.         }]
129.     }, {
130.         value: 'notice',
131.         label: 'Notice',
132.         children: [{
133.             value: 'alert',
134.             label: 'Alert 警告'
135.         }, {
136.             value: 'loading',
137.             label: 'Loading 加载'
138.         }, {
139.             value: 'message',
140.             label: 'Message 消息提示'
141.         }, {
142.             value: 'message-box',
143.             label: 'MessageBox 弹框'
144.         }, {
145.             value: 'notification',
146.             label: 'Notification 通知'
147.         }]
148.     }, {
149.         value: 'navigation',
150.         label: 'Navigation',
```

```

151.         children: [{
152.             value: 'menu',
153.             label: 'NavMenu 导航菜单'
154.         }, {
155.             value: 'tabs',
156.             label: 'Tabs 标签页'
157.         }, {
158.             value: 'breadcrumb',
159.             label: 'Breadcrumb 面包屑'
160.         }, {
161.             value: 'dropdown',
162.             label: 'Dropdown 下拉菜单'
163.         }, {
164.             value: 'steps',
165.             label: 'Steps 步骤条'
166.         }]
167.     }, {
168.         value: 'others',
169.         label: 'Others',
170.         children: [{
171.             value: 'dialog',
172.             label: 'Dialog 对话框'
173.         }, {
174.             value: 'tooltip',
175.             label: 'Tooltip 文字提示'
176.         }, {
177.             value: 'popover',
178.             label: 'Popover 弹出框'
179.         }, {
180.             value: 'card',
181.             label: 'Card 卡片'
182.         }, {
183.             value: 'carousel',
184.             label: 'Carousel 走马灯'
185.         }, {
186.             value: 'collapse',
187.             label: 'Collapse 折叠面板'
188.         }]

```

```

189.         ]]
190.     }, {
191.         value: 'ziyuan',
192.         label: '资源',
193.         children: [{
194.             value: 'axure',
195.             label: 'Axure Components'
196.         }, {
197.             value: 'sketch',
198.             label: 'Sketch Templates'
199.         }, {
200.             value: 'jiaohu',
201.             label: '组件交互文档'
202.         }
203.     ]],
204.     selectedOptions3: ['zujian', 'data', 'tag']
205. };
206. }
207. };
208. </script>

```

## 选择即改变

点击或移入选项即表示选中该项，可用于选择任意一级菜单的选项。



若需要允许用户选择任意一级选项，则可将 `change-on-select` 赋值

为 `true`

```

1. <el-cascader
2.   :options="options"
3.   change-on-select
4. ></el-cascader>
5. <script>
6.   export default {
7.     data() {
8.       return {
9.         options: [{
10.          value: 'zhinan',
11.          label: '指南',
12.          children: [{
13.            value: 'shejiyuanze',
14.            label: '设计原则',
15.            children: [{
16.              value: 'yizhi',
17.              label: '一致'
18.            }, {
19.              value: 'fankui',
20.              label: '反馈'
21.            }, {
22.              value: 'xiaolv',
23.              label: '效率'
24.            }, {
25.              value: 'kekong',
26.              label: '可控'
27.            }
28.          ], {
29.            value: 'daohang',
30.            label: '导航',
31.            children: [{
32.              value: 'cexiangdaohang',
33.              label: '侧向导航'
34.            }, {
35.              value: 'dingbudaohang',
36.              label: '顶部导航'

```

```
37.         }]  
38.     }]  
39.     }, {  
40.         value: 'zujian',  
41.         label: '组件',  
42.         children: [{  
43.             value: 'basic',  
44.             label: 'Basic',  
45.             children: [{  
46.                 value: 'layout',  
47.                 label: 'Layout 布局'  
48.             }, {  
49.                 value: 'color',  
50.                 label: 'Color 色彩'  
51.             }, {  
52.                 value: 'typography',  
53.                 label: 'Typography 字体'  
54.             }, {  
55.                 value: 'icon',  
56.                 label: 'Icon 图标'  
57.             }, {  
58.                 value: 'button',  
59.                 label: 'Button 按钮'  
60.             }]  
61.         }, {  
62.             value: 'form',  
63.             label: 'Form',  
64.             children: [{  
65.                 value: 'radio',  
66.                 label: 'Radio 单选框'  
67.             }, {  
68.                 value: 'checkbox',  
69.                 label: 'Checkbox 多选框'  
70.             }, {  
71.                 value: 'input',  
72.                 label: 'Input 输入框'  
73.             }, {  
74.                 value: 'input-number',
```



```

75.         label: 'InputNumber 计数器'
76.     }, {
77.         value: 'select',
78.         label: 'Select 选择器'
79.     }, {
80.         value: 'cascader',
81.         label: 'Cascader 级联选择器'
82.     }, {
83.         value: 'switch',
84.         label: 'Switch 开关'
85.     }, {
86.         value: 'slider',
87.         label: 'Slider 滑块'
88.     }, {
89.         value: 'time-picker',
90.         label: 'TimePicker 时间选择器'
91.     }, {
92.         value: 'date-picker',
93.         label: 'DatePicker 日期选择器'
94.     }, {
95.         value: 'datetime-picker',
96.         label: 'DateTimePicker 日期时间选择器'
97.     }, {
98.         value: 'upload',
99.         label: 'Upload 上传'
100.    }, {
101.        value: 'rate',
102.        label: 'Rate 评分'
103.    }, {
104.        value: 'form',
105.        label: 'Form 表单'
106.    }]
107. }, {
108.     value: 'data',
109.     label: 'Data',
110.     children: [{
111.         value: 'table',
112.         label: 'Table 表格'

```

```
113.         }, {
114.             value: 'tag',
115.             label: 'Tag 标签'
116.         }, {
117.             value: 'progress',
118.             label: 'Progress 进度条'
119.         }, {
120.             value: 'tree',
121.             label: 'Tree 树形控件'
122.         }, {
123.             value: 'pagination',
124.             label: 'Pagination 分页'
125.         }, {
126.             value: 'badge',
127.             label: 'Badge 标记'
128.         }]
129.     }, {
130.         value: 'notice',
131.         label: 'Notice',
132.         children: [{
133.             value: 'alert',
134.             label: 'Alert 警告'
135.         }, {
136.             value: 'loading',
137.             label: 'Loading 加载'
138.         }, {
139.             value: 'message',
140.             label: 'Message 消息提示'
141.         }, {
142.             value: 'message-box',
143.             label: 'MessageBox 弹框'
144.         }, {
145.             value: 'notification',
146.             label: 'Notification 通知'
147.         }]
148.     }, {
149.         value: 'navigation',
150.         label: 'Navigation',
```

```
151.         children: [{
152.             value: 'menu',
153.             label: 'NavMenu 导航菜单'
154.         }, {
155.             value: 'tabs',
156.             label: 'Tabs 标签页'
157.         }, {
158.             value: 'breadcrumb',
159.             label: 'Breadcrumb 面包屑'
160.         }, {
161.             value: 'dropdown',
162.             label: 'Dropdown 下拉菜单'
163.         }, {
164.             value: 'steps',
165.             label: 'Steps 步骤条'
166.         }]
167.     }, {
168.         value: 'others',
169.         label: 'Others',
170.         children: [{
171.             value: 'dialog',
172.             label: 'Dialog 对话框'
173.         }, {
174.             value: 'tooltip',
175.             label: 'Tooltip 文字提示'
176.         }, {
177.             value: 'popover',
178.             label: 'Popover 弹出框'
179.         }, {
180.             value: 'card',
181.             label: 'Card 卡片'
182.         }, {
183.             value: 'carousel',
184.             label: 'Carousel 走马灯'
185.         }, {
186.             value: 'collapse',
187.             label: 'Collapse 折叠面板'
188.         }]
```

```

189.         ]]
190.     }, {
191.         value: 'ziyuan',
192.         label: '资源',
193.         children: [{
194.             value: 'axure',
195.             label: 'Axure Components'
196.         }, {
197.             value: 'sketch',
198.             label: 'Sketch Templates'
199.         }, {
200.             value: 'jiaohu',
201.             label: '组件交互文档'
202.         }]
203.     }]
204. };
205. }
206. };
207. </script>

```

## 动态加载次级选项

当选中某一级时，动态加载该级下的选项。



本例的选项数据源在初始化时不包含城市数据。利用active-item-

change事件，可以在用户点击某个省份时拉取该省份下的城市数据。此外，本例还展示了props属性的用法。

```
1. <el-cascader
2.   :options="options2"
3.   @active-item-change="handleItemChange"
4.   :props="props"
5. ></el-cascader>
6.
7. <script>
8.   export default {
9.     data() {
10.      return {
11.        options2: [{
12.          label: '江苏',
13.          cities: []
14.        }, {
15.          label: '浙江',
16.          cities: []
17.        }],
18.        props: {
19.          value: 'label',
20.          children: 'cities'
21.        }
22.      };
23.    },
24.
25.    methods: {
26.      handleItemChange(val) {
27.        console.log('active item:', val);
28.        setTimeout(_ => {
29.          if (val.indexOf('江苏') > -1 &&
!this.options2[0].cities.length) {
30.            this.options2[0].cities = [{
31.              label: '南京'
32.            }];
33.          } else if (val.indexOf('浙江') > -1 &&
!this.options2[1].cities.length) {
```

```

34.         this.options2[1].cities = [{
35.             label: '杭州'
36.         }];
37.     }
38.     }, 300);
39. }
40. }
41. };
42. </script>

```

## 可搜索

可以快捷地搜索选项并选择。

只可选择最后一级菜单的选项

试试搜索：指南

可选择任意一级菜单的选项

试试搜索：指南

将 `filterable` 赋值为 `true` 即可打开搜索功能。

```

1. <div class="block">
2.   <span class="demonstration">只可选择最后一级菜单的选项</span>
3.   <el-cascader
4.     placeholder="试试搜索：指南"
5.     :options="options"
6.     filterable
7.   ></el-cascader>
8. </div>
9. <div class="block">
10.  <span class="demonstration">可选择任意一级菜单的选项</span>
11.  <el-cascader
12.    placeholder="试试搜索：指南"
13.    :options="options"
14.    filterable
15.    change-on-select
16.  ></el-cascader>
17. </div>

```

```
18.  
19. <script>  
20.   export default {  
21.     data() {  
22.       return {  
23.         options: [{  
24.           value: 'zhinan',  
25.           label: '指南',  
26.           children: [{  
27.             value: 'shejiyuanze',  
28.             label: '设计原则',  
29.             children: [{  
30.               value: 'yizhi',  
31.               label: '一致'  
32.             }, {  
33.               value: 'fankui',  
34.               label: '反馈'  
35.             }, {  
36.               value: 'xiaolv',  
37.               label: '效率'  
38.             }, {  
39.               value: 'kekong',  
40.               label: '可控'  
41.             }]  
42.           }, {  
43.             value: 'daohang',  
44.             label: '导航',  
45.             children: [{  
46.               value: 'cexiangdaohang',  
47.               label: '侧向导航'  
48.             }, {  
49.               value: 'dingbudaohang',  
50.               label: '顶部导航'  
51.             }]  
52.           }]  
53.         }, {  
54.           value: 'zujian',  
55.           label: '组件',
```

```
56.         children: [{
57.             value: 'basic',
58.             label: 'Basic',
59.             children: [{
60.                 value: 'layout',
61.                 label: 'Layout 布局'
62.             }, {
63.                 value: 'color',
64.                 label: 'Color 色彩'
65.             }, {
66.                 value: 'typography',
67.                 label: 'Typography 字体'
68.             }, {
69.                 value: 'icon',
70.                 label: 'Icon 图标'
71.             }, {
72.                 value: 'button',
73.                 label: 'Button 按钮'
74.             }]
75.         }, {
76.             value: 'form',
77.             label: 'Form',
78.             children: [{
79.                 value: 'radio',
80.                 label: 'Radio 单选框'
81.             }, {
82.                 value: 'checkbox',
83.                 label: 'Checkbox 多选框'
84.             }, {
85.                 value: 'input',
86.                 label: 'Input 输入框'
87.             }, {
88.                 value: 'input-number',
89.                 label: 'InputNumber 计数器'
90.             }, {
91.                 value: 'select',
92.                 label: 'Select 选择器'
93.             }, {
```



```
94.         value: 'cascader',
95.         label: 'Cascader 级联选择器'
96.     }, {
97.         value: 'switch',
98.         label: 'Switch 开关'
99.     }, {
100.        value: 'slider',
101.        label: 'Slider 滑块'
102.    }, {
103.        value: 'time-picker',
104.        label: 'TimePicker 时间选择器'
105.    }, {
106.        value: 'date-picker',
107.        label: 'DatePicker 日期选择器'
108.    }, {
109.        value: 'datetime-picker',
110.        label: 'DateTimePicker 日期时间选择器'
111.    }, {
112.        value: 'upload',
113.        label: 'Upload 上传'
114.    }, {
115.        value: 'rate',
116.        label: 'Rate 评分'
117.    }, {
118.        value: 'form',
119.        label: 'Form 表单'
120.    }
121. ], {
122.     value: 'data',
123.     label: 'Data',
124.     children: [{
125.         value: 'table',
126.         label: 'Table 表格'
127.     }, {
128.         value: 'tag',
129.         label: 'Tag 标签'
130.     }, {
131.         value: 'progress',
```

```
132.         label: 'Progress 进度条'
133.     }, {
134.         value: 'tree',
135.         label: 'Tree 树形控件'
136.     }, {
137.         value: 'pagination',
138.         label: 'Pagination 分页'
139.     }, {
140.         value: 'badge',
141.         label: 'Badge 标记'
142.     }]
143. }, {
144.     value: 'notice',
145.     label: 'Notice',
146.     children: [{
147.         value: 'alert',
148.         label: 'Alert 警告'
149.     }, {
150.         value: 'loading',
151.         label: 'Loading 加载'
152.     }, {
153.         value: 'message',
154.         label: 'Message 消息提示'
155.     }, {
156.         value: 'message-box',
157.         label: 'MessageBox 弹框'
158.     }, {
159.         value: 'notification',
160.         label: 'Notification 通知'
161.     }]
162. }, {
163.     value: 'navigation',
164.     label: 'Navigation',
165.     children: [{
166.         value: 'menu',
167.         label: 'NavMenu 导航菜单'
168.     }, {
169.         value: 'tabs',
```

```
170.         label: 'Tabs 标签页'
171.     }, {
172.         value: 'breadcrumb',
173.         label: 'Breadcrumb 面包屑'
174.     }, {
175.         value: 'dropdown',
176.         label: 'Dropdown 下拉菜单'
177.     }, {
178.         value: 'steps',
179.         label: 'Steps 步骤条'
180.     }]
181. }, {
182.     value: 'others',
183.     label: 'Others',
184.     children: [{
185.         value: 'dialog',
186.         label: 'Dialog 对话框'
187.     }, {
188.         value: 'tooltip',
189.         label: 'Tooltip 文字提示'
190.     }, {
191.         value: 'popover',
192.         label: 'Popover 弹出框'
193.     }, {
194.         value: 'card',
195.         label: 'Card 卡片'
196.     }, {
197.         value: 'carousel',
198.         label: 'Carousel 走马灯'
199.     }, {
200.         value: 'collapse',
201.         label: 'Collapse 折叠面板'
202.     }]
203. }]
204. }, {
205.     value: 'ziyuan',
206.     label: '资源',
207.     children: [{
```

```

208.         value: 'axure',
209.         label: 'Axure Components'
210.     }, {
211.         value: 'sketch',
212.         label: 'Sketch Templates'
213.     }, {
214.         value: 'jiaohu',
215.         label: '组件交互文档'
216.     }]
217.   ]
218. };
219. }
220. };
221. </script>

```

## Attributes

参数	说明	类型	可选值	默认值
options	可选项数据源，键名可通过 props 属性配置	array	—	—
props	配置选项，具体见下表	object	—	—
value	选中项绑定值	array	—	—
separator	选项分隔符	string	—	斜杠 '/'
popper-class	自定义浮层类名	string	—	—
placeholder	输入框占位文本	string	—	请选择
disabled	是否禁用	boolean	—	false
clearable	是否支持清空选项	boolean	—	false
expand-trigger	次级菜单的展开方式	string	click / hover	click
show-all-levels	输入框中是否显示选中值的完整路径	boolean	—	true
filterable	是否可搜索选项	boolean	—	—
debounce	搜索关键词输入的去抖延迟，毫秒	number	—	300
change-on-	是否允许选择任意一级的选	boolean	—	false

select	项	boolean	–	false
size	尺寸	string	medium / small / mini	–
before-filter	筛选之前的钩子，参数为输入的值，若返回 false 或者返回 Promise 且被 reject，则停止筛选	function(value)	–	–

## props

参数 说明 类型 可选值 默认值

value 指定选项的值为选项对象的某个属性值 string – –

label 指定选项标签为选项对象的某个属性值 string – –

children 指定选项的子选项为选项对象的某个属性值 string – –

disabled 指定选项的禁用为选项对象的某个属性值 string – –

## Events

事件名称	说明	回调参数
change	当绑定值变化时触发的事件	当前值
active-item-change	当父级选项变化时触发的事件，仅在 change-on-select 为 false 时可用	各父级选项组成的数组

## 3.7 Switch 开关

- Switch 开关
  - 基本用法
  - 文字描述
  - 扩展的 value 类型
  - 禁用状态
  - Attributes
  - Events
  - Methods

### Switch 开关

表示两种相互对立的状态间的切换，多用于触发「开/关」。

### 基本用法



绑定 `v-model` 到一个 `Boolean` 类型的变量。可以使用 `active-color` 属性与 `inactive-color` 属性来设置开关的背景色。

```
1. <el-switch
2.   v-model="value2"
3.   active-color="#13ce66"
4.   inactive-color="#ff4949">
5. </el-switch>
6.
7. <script>
8.   export default {
9.     data() {
10.       return {
```

```

11.         value1: true,
12.         value2: true
13.     }
14. }
15. };
16. </script>

```

## 文字描述

按年付费  按月付费

按年付费  按月付费

使用 `active-text` 属性与 `inactive-text` 属性来设置开关的文字描述。

```

1. <el-switch
2.   v-model="value3"
3.   active-text="按月付费"
4.   inactive-text="按年付费">
5. </el-switch>
6. <el-switch
7.   style="display: block"
8.   v-model="value4"
9.   active-color="#13ce66"
10.  inactive-color="#ff4949"
11.  active-text="按月付费"
12.  inactive-text="按年付费">
13. </el-switch>
14.
15. <script>
16.   export default {
17.     data() {
18.       return {
19.         value3: true,
20.         value4: true
21.       }

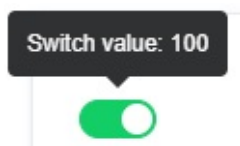
```

```

22.     }
23.   };
24. </script>

```

## 扩展的 value 类型



设置 `active-value` 和 `inactive-value` 属性，接受 `Boolean`，`String` 或 `Number` 类型的值。

```

1. <el-tooltip :content="'Switch value: ' + value5" placement="top">
2.   <el-switch
3.     v-model="value5"
4.     active-color="#13ce66"
5.     inactive-color="#ff4949"
6.     active-value="100"
7.     inactive-value="0">
8.   </el-switch>
9. </el-tooltip>
10.
11. <script>
12.   export default {
13.     data() {
14.       return {
15.         value5: '100'
16.       }
17.     }
18.   };
19. </script>

```

## 禁用状态





设置 `disabled` 属性，接受一个 `Boolean`，设置 `true` 即可禁用。

```
1. <el-switch
2.   v-model="value6"
3.   disabled>
4. </el-switch>
5. <el-switch
6.   v-model="value7"
7.   disabled>
8. </el-switch>
9. <script>
10.   export default {
11.     data() {
12.       return {
13.         value6: true,
14.         value7: false
15.       }
16.     }
17.   };
18. </script>
```

## Attributes

参数	说明	类型	可选值	默认值
disabled	是否禁用	boolean	—	false
width	switch 的宽度（像素）	number	—	40
active-icon-class	switch 打开时所显示图标类名，设置此项会忽略 active-text	string	—	—
inactive-icon-class	switch 关闭时所显示图标类名，设置此项会忽略 inactive-text	string	—	—
active-text	switch 打开时的文字描述	string	—	—

inactive-text	switch 关闭时的文字描述	string	—	—
active-value	switch 打开时的值	boolean / string / number	—	true
inactive-value	switch 关闭时的值	boolean / string / number	—	false
active-color	switch 打开时的背景色	string	—	#409EFF
inactive-color	switch 关闭时的背景色	string	—	#C0CCDA
name	switch 对应的 name 属性	string	—	—

## Events

事件名称	说明	回调参数
change	switch 状态发生变化时的回调函数	新状态的值

## Methods

| 方法名 | 说明 | 参数 |

| focus | 使 Switch 获取焦点 | - |

## 3.8 Slider 滑块

- Slider 滑块
  - 基础用法
  - 离散值
  - 带有输入框
  - 范围选择
  - 竖向模式
  - Attributes
  - Events

## Slider 滑块

通过拖动滑块在一个固定区间内进行选择

## 基础用法

在拖动滑块时，显示当前值



## 通过设置绑定值自定义滑块的初始值

```
1. <template>
2.   <div class="block">
3.     <span class="demonstration">默认</span>
4.     <el-slider v-model="value1"></el-slider>
5.   </div>
6.   <div class="block">
7.     <span class="demonstration">自定义初始值</span>
8.     <el-slider v-model="value2"></el-slider>
9.   </div>
10.  <div class="block">
11.    <span class="demonstration">隐藏 Tooltip</span>
12.    <el-slider v-model="value3" :show-tooltip="false"></el-slider>
13.  </div>
14.  <div class="block">
15.    <span class="demonstration">格式化 Tooltip</span>
16.    <el-slider v-model="value4" :format-tooltip="formatTooltip">
17.      </el-slider>
18.  </div>
19.  <div class="block">
20.    <span class="demonstration">禁用</span>
21.    <el-slider v-model="value5" disabled></el-slider>
22.  </div>
23. </template>
24. <script>
25.   export default {
26.     data() {
27.       return {
28.         value1: 0,
29.         value2: 50,
30.         value3: 36,
31.         value4: 48,
32.         value5: 42
33.       }
34.     },
35.     methods: {
```

```

36.     formatTooltip(val) {
37.         return val / 100;
38.     }
39. }
40. }
41. </script>

```

## 离散值

选项可以是离散的

不显示间断点



显示间断点



改变 `step` 的值可以改变步长，通过设置 `show-step` 属性可以显示间断点

```

1. <template>
2.   <div class="block">
3.     <span class="demonstration">不显示间断点</span>
4.     <el-slider
5.       v-model="value6"
6.       :step="10">
7.     </el-slider>
8.   </div>
9.   <div class="block">
10.    <span class="demonstration">显示间断点</span>
11.    <el-slider
12.      v-model="value7"
13.      :step="10"
14.      show-stops>
15.    </el-slider>
16.  </div>
17. </template>
18.

```

```

19. <script>
20.   export default {
21.     data() {
22.       return {
23.         value6: 0,
24.         value7: 0
25.       }
26.     }
27.   }
28. </script>

```

## 带有输入框

通过输入框设置精确数值



设置 `show-input` 属性会在右侧显示一个输入框

```

1. <template>
2.   <div class="block">
3.     <el-slider
4.       v-model="value8"
5.       show-input>
6.     </el-slider>
7.   </div>
8. </template>
9.
10. <script>
11.   export default {
12.     data() {
13.       return {
14.         value8: 0
15.       }
16.     }
17.   }

```

```
18. </script>
```

## 范围选择

支持选择某一数值范围



设置 `range` 即可开启范围选择，此时绑定值是一个数组，其元素分别为最小边界值和最大边界值

```
1. <template>
2.   <div class="block">
3.     <el-slider
4.       v-model="value9"
5.       range
6.       show-stops
7.       :max="10">
8.     </el-slider>
9.   </div>
10. </template>
11.
12. <script>
13.   export default {
14.     data() {
15.       return {
16.         value9: [4, 8]
17.       }
18.     }
19.   }
20. </script>
```

## 竖向模式



设置 `vertical` 可使 Slider 变成竖向模式，此时必须设置高度 `height` 属性

```
1. <template>
2.   <div class="block">
3.     <el-slider
4.       v-model="value10"
5.       vertical
6.       height="200px">
7.     </el-slider>
8.   </div>
9. </template>
10.
11. <script>
12.   export default {
13.     data() {
14.       return {
15.         value10: 0
16.       }
17.     }
18.   }
19. </script>
```

## Attributes

参数	说明	类型	可选值	默认值
----	----	----	-----	-----



min	最小值	number	—	0
max	最大值	number	—	100
disabled	是否禁用	boolean	—	false
step	步长	number	—	1
show-input	是否显示输入框，仅在非范围选择时有效	boolean	—	false
show-input-controls	在显示输入框的情况下，是否显示输入框的控制按钮	boolean	—	true
show-stops	是否显示间断点	boolean	—	false
show-tooltip	是否显示 tooltip	boolean	—	true
format-tooltip	格式化 tooltip message	function(value)	—	—
range	是否为范围选择	boolean	—	false
vertical	是否竖向模式	boolean	—	false
height	Slider 高度，竖向模式时必填	string	—	—
label	屏幕阅读器标签	string	—	—
debounce	输入时的去抖延迟，毫秒，仅在 show-input 等于 true 时有效	number	—	300

## Events

事件名称	说明	回调参数
change	值改变时触发（使用鼠标拖曳时，只在松开鼠标后触发）	改变后的值

## 3.9 TimePicker 时间选择器