Pagination

If you have too much data to display in one page, use pagination.

Basic usage

:::demo Set `layout` with different pagination elements you wish to display separated with a comma. Pagination elements are: `prev` (a button navigating to the previous page), `next` (a button navigating to the next page), `pager` (page list), `jumper` (a jump-to input), `total` (total item count), `size` (a select to determine page size) and `->` (every element after this symbol will be pulled to the right).

```html
<div class="block">
  <span class="demonstration">When you have few pages</span>
  <el-pagination
    layout="prev, pager, next"
    :total="50">
  </el-pagination>
</div>
<div class="block">
  <span class="demonstration">When you have more than 7 pages</span>
  <el-pagination
    layout="prev, pager, next"
    :total="1000">
  </el-pagination>
</div>
```

:::

Number of pagers

:::demo By default, Pagination collapses extra pager buttons when it has more than 7 pages. This can be configured with the `pager-count` attribute.

```html
<el-pagination
  :page-size="20"
  :pager-count="11"
  layout="prev, pager, next"
  :total="1000">
</el-pagination>
```

:::

Buttons with background color

:::demo Set the `background` attribute and the buttons will have a background color.

```html
<el-pagination
  background
  layout="prev, pager, next"
  :total="1000">
</el-pagination>
```

:::

Small Pagination

Use small pagination in the case of limited space.

:::demo Just set the `small` attribute to `true` and the Pagination becomes smaller.

```html
<el-pagination
  small
  layout="prev, pager, next"
  :total="50">
</el-pagination>
```

:::

More elements

Add more modules based on your scenario.

:::demo This example is a complete use case. It uses `size-change` and `current-change` event to handle page size changes and current page changes. `page-sizes` accepts an array of integers, each of which represents a different page size in the `sizes` select options, e.g. `[100, 200, 300, 400]` indicates that the select will have four options: 100, 200, 300 or 400 items per page.

```html
<template>
  <div class="block">
    <span class="demonstration">Total item count</span>
    <el-pagination
      @size-change="handleSizeChange"
      @current-change="handleCurrentChange"
      :current-page.sync="currentPage1"
      :page-size="100"
      layout="total, prev, pager, next"
      :total="1000">
    </el-pagination>
  </div>
  <div class="block">
    <span class="demonstration">Change page size</span>
    <el-pagination
      @size-change="handleSizeChange"
      @current-change="handleCurrentChange"
      :current-page.sync="currentPage2"
      :page-sizes="[100, 200, 300, 400]"
      :page-size="100"
      layout="sizes, prev, pager, next"
      :total="1000">
    </el-pagination>
  </div>
  <div class="block">
    <span class="demonstration">Jump to</span>
    <el-pagination
      @size-change="handleSizeChange"
      @current-change="handleCurrentChange"
      :current-page.sync="currentPage3"
      :page-size="100"
      layout="prev, pager, next, jumper"
      :total="1000">
    </el-pagination>
  </div>
  <div class="block">
    <span class="demonstration">All combined</span>
    <el-pagination
      @size-change="handleSizeChange"
      @current-change="handleCurrentChange"
      :current-page.sync="currentPage4"
      :page-sizes="[100, 200, 300, 400]"
      :page-size="100"
      layout="total, sizes, prev, pager, next, jumper"
      :total="400">
    </el-pagination>
  </div>
</template>
<script>
  export default {
    methods: {
      handleSizeChange(val) {
        console.log(`${val} items per page`);
      },
      handleCurrentChange(val) {
        console.log(`current page: ${val}`);
      }
    },
    data() {
      return {
        currentPage1: 5,
        currentPage2: 5,
        currentPage3: 5,
        currentPage4: 4
      };
    }
  }
</script>
```

:::

Hide pagination when there is only one page

When there is only one page, hide the pagination by setting the `hide-on-single-page` attribute.

:::demo

```
<div>
 <el-switch v-model="value">
 </el-switch>
 <el-pagination
  :hide-on-single-page="value"
  :total="5"
  layout="prev, pager, next">
</el-pagination>
</div>

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

:::

Attributes

Attribute	Description	Type	Accepted Values	Default
small	whether to use small pagination	boolean	—	false
background	whether the buttons have a background color	boolean	—	false
page-size	item count of each page, supports the .sync modifier	number	—	10
total	total item count	number	—	—
page-count	total page count. Set either `total` or `page-count` and pages will be displayed; if you need `page-sizes`, `total` is required	number	—	—
pager-count	number of pagers. Pagination collapses when the total page count exceeds this value	number	odd number between 5 and 21	7
current-page	current page number, supports the .sync modifier	number	—	1
layout	layout of Pagination, elements separated with a comma	string	`sizes`, `prev`, `pager`, `next`, `jumper`, `->`, `total`, `slot`	'prev, pager, next, jumper, ->, total'
page-sizes	options of item count per page	number[]	—	[10, 20,

				30, 40, 50, 100]
popper-class	custom class name for the page size Select's dropdown	string	—	—
prev-text	text for the prev button	string	—	—
next-text	text for the next button	string	—	—
disabled	whether Pagination is disabled	boolean	—	false
hide-on-single-page	whether to hide when there's only one page	boolean	—	-

Events

Event Name	Description	Parameters
size-change	triggers when `page-size` changes	the new page size
current-change	triggers when `current-page` changes	the new current page
prev-click	triggers when the prev button is clicked and current page changes	the new current page
next-click	triggers when the next button is clicked and current page changes	the new current page

Slot

Name	Description
—	custom content. To use this, you need to declare `slot` in `layout`