Skeleton

When loading data, and you need a rich experience for visual and interactions for your end users, you can choose `skeleton`.

Basic usage

The basic skeleton.

:::demo

```
<template>
  <el-skeleton />
</template>
```

:::

Configurable Rows

You can configure the row numbers yourself, we are rendering a title row with 33% width of the others.

:::demo

```
<el-skeleton :rows="6" />
```

:::

Animation

We have provided a switch flag indicating whether showing the loading animation, called `animated` when this is true, all children of `el-skeleton` will show animation

:::demo

```
<el-skeleton :rows="6" animated />
```

:::

Customized Template

Element only provides the most common template, sometimes that could be a problem, so you have a slot named `template` to do that work.

Also we have provided different types skeleton unit that you can choose, for more detailed info, please scroll down to the bottom of this page to see the API description. Also, when building your own customized skeleton structure, you should be structuring them as closer to the real DOM as possible, which avoiding the DOM bouncing caused by the height difference.

:::demo

```
<template>
  <el-skeleton style="width: 240px">
    <template slot="template">
```

```
        <el-skeleton-item variant="image" style="width: 240px; height: 240px;" />
        <div style="padding: 14px;">
          <el-skeleton-item variant="p" style="width: 50%" />
          <div
            style="display: flex; align-items: center; justify-items: space-between;"
          >
            <el-skeleton-item variant="text" style="margin-right: 16px;" />
            <el-skeleton-item variant="text" style="width: 30%;" />
          </div>
        </div>
      </template>
    </el-skeleton>
  </template>
```

:::

Loading state

When `Loading` ends, we always need to show the real UI with data to our end users. with the attribtue `loading` we can control whether showing the DOM. You can also use slot `default` to structure the real DOM element.

:::demo

```
<template>
  <div style="width: 240px">
    <p>
      <label style="margin-right: 16px;">Switch Loading</label>
      <el-switch v-model="loading" />
    </p>
    <el-skeleton style="width: 240px" :loading="loading" animated>
      <template slot="template">
        <el-skeleton-item
          variant="image"
          style="width: 240px; height: 240px;"
        />
        <div style="padding: 14px;">
          <el-skeleton-item variant="h3" style="width: 50%;" />
          <div
            style="display: flex; align-items: center; justify-items: space-between;
margin-top: 16px; height: 16px;"
          >
            <el-skeleton-item variant="text" style="margin-right: 16px;" />
            <el-skeleton-item variant="text" style="width: 30%;" />
          </div>
        </div>
      </template>
      <template>
        <el-card :body-style="{ padding: '0px', marginBottom: '1px' }">
          <img
            src="https://shadow.elemecdn.com/app/element/hamburger.9cf7b091-55e9-
11e9-a976-7f4d0b07eef6.png"
            class="image"
          />
          <div style="padding: 14px;">
            <span>Delicious hamberger</span>
            <div class="bottom card-header">
              <span class="time">{{ currentDate }}</span>
              <el-button type="text" class="button">Operation button</el-button>
            </div>
          </div>
        </el-card>
      </template>
    </el-skeleton>
  </div>
</template>

<script>
  export default {
    data () {
      return {
        loading: true,
        currentDate: '2021-06-01'
      }
    },
  }
</script>
```

:::

Rendering a list of data

Most of the time, skeleton is used as indicators of rendering a list of data which haven't been fetched from server yet, then we need to create a list of skeleton out of no where to make it look like it is loading, with `count` attribute, you can control how many these templates you need to render to the browser.

:::tip We do not recommend rendering lots of fake UI to the browser, it will still cause the performance issue, it also costs longer to destroy the skeleton. Keep `count` as small as it can be to make better user experience. :::

:::demo

```
<template>
  <div style="width: 400px">
    <p>
      <el-button @click="setLoading">Click me to reload</el-button>
    </p>
    <el-skeleton style="width:400px" :loading="loading" animated :count="3">
      <template slot="template">
        <el-skeleton-item
          variant="image"
          style="width: 400px; height: 267px;"
        />
        <div style="padding: 14px;">
          <el-skeleton-item variant="h3" style="width: 50%;" />
          <div
```

```
            style="display: flex; align-items: center; justify-items: space-between;
margin-top: 16px; height: 16px;"
          >
            <el-skeleton-item variant="text" style="margin-right: 16px;" />
            <el-skeleton-item variant="text" style="width: 30%;" />
          </div>
        </div>
      </template>
      <template>
        <el-card
          :body-style="{ padding: '0px', marginBottom: '1px' }"
          v-for="item in lists"
          :key="item.name"
        >
          <img :src="item.imgUrl" class="image multi-content" />
          <div style="padding: 14px;">
            <span>Delicious hamberger</span>
            <div class="bottom card-header">
              <span class="time">{{ currentDate }}</span>
              <el-button type="text" class="button">Operation button</el-button>
            </div>
          </div>
        </el-card>
      </template>
    </el-skeleton>
  </div>
</template>

<script>
  export default {
    data() {
      return {
        loading: true,
        currentDate: '2021-06-01',
        lists: [],
      }
    },
    mounted() {
      this.loading = false
      this.lists = [
        {
          imgUrl:

'https://fuss10.elemecdn.com/a/3f/3302e58f9a181d2509f3dc0fa68b0jpeg.jpeg',
          name: 'Deer',
        },
        {
          imgUrl:

'https://fuss10.elemecdn.com/1/34/19aa98b1fcb2781c4fba33d850549jpeg.jpeg',
          name: 'Horse',
        },
        {
          imgUrl:

'https://fuss10.elemecdn.com/0/6f/e35ff375812e6b0020b6b4e8f9583jpeg.jpeg',
          name: 'Mountain Lion',
        },
      ]
    },
    methods: {
      setLoading() {
        this.loading = true
        setTimeout(() => (this.loading = false), 2000)
      },
    },
  }
</script>
```

:::

Avoiding rendering bouncing.

Sometimes API responds very quickly, when that happens, the skeleton just gets rendered to the DOM then it needs to switch back to real DOM, that causes the sudden flashy. To avoid such thing, you can use the `throttle` attribute.

:::demo

```
<template>
  <div style="width: 240px">
    <p>
      <label style="margin-right: 16px;">Switch Loading</label>
      <el-switch v-model="loading" />
    </p>
    <el-skeleton
      style="width: 240px"
      :loading="loading"
      animated
      :throttle="500"
    >
      <template slot="template">
        <el-skeleton-item
          variant="image"
          style="width: 240px; height: 240px;"
        />
        <div style="padding: 14px;">
          <el-skeleton-item variant="h3" style="width: 50%;" />
          <div
            style="display: flex; align-items: center; justify-items: space-between;
margin-top: 16px; height: 16px;"
          >
            <el-skeleton-item variant="text" style="margin-right: 16px;" />
            <el-skeleton-item variant="text" style="width: 30%;" />
          </div>
        </div>
      </template>
      <template>
        <el-card :body-style="{ padding: '0px', marginBottom: '1px'}">
          <img
            src="https://shadow.elemecdn.com/app/element/hamburger.9cf7b091-55e9-
11e9-a976-7f4d0b07eef6.png"
            class="image"
          />
          <div style="padding: 14px;">
            <span>Delicious hamberger</span>
            <div class="bottom card-header">
              <span class="time">{{ currentDate }}</span>
              <el-button type="text" class="button">operation button</el-button>
            </div>
          </div>
        </el-card>
      </template>
    </el-skeleton>
  </div>
</template>

<script>
  export default {
    data() {
      return {
        loading: false,
        currentDate: '2021-06-01'
      }
    },
  }
</script>
```

:::

Skeleton Attributes

Attribute	Description	Type	Acceptable Value	Default
animated	whether showing the animation	boolean	true / false	false
count	how many fake items to render to the DOM	number	integer	1
loading	whether showing the skeleton	boolean	true / false	true
rows	numbers of the row, only useful when no template slot were given	number	integer	4
throttle	Rendering delay in millseconds	number	integer	0

Skeleton Item Attributes

Attribute	Description	Type	Acceptable Value	Default
variant	The current rendering skeleton type	Enum(string)	p / text / h1 / h3 / text / caption / button / image / circle / rect	text

Skeleton Slots

Name	Description
default	Real rendering DOM
template	Custom rendering skeleton template