# Skeleton 骨架屏

在需要等待加载内容的位置设置一个骨架屏，某些场景下比 Loading 的视觉效果更好。

## 基础用法

基础的骨架效果。

:::demo

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

:::

## 更多参数

可以配置骨架屏段落数量，以便更接近真实渲染效果。首行会被渲染一个长度 33% 的段首。

:::demo

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

:::

## 动画效果

显示动画效果。

:::demo

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

:::

## 自定义样式

Element 提供的排版模式有时候并不满足要求，当您想要用自己定义的模板时，可以通过一个具名 Slot 来自己设定模板。

我们提供了不同的模板单元可供使用，具体可选值请看 API 详细描述。 建议在描述模板的时候，尽量靠近真实的 DOM 结构，这样可以避免 DOM 高度差距引起的抖动。 :::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 状态

当 Loading 结束之后，我们往往需要显示真实的 UI，可以通过 `loading` 的值来控制是否显示真实的 DOM。然后通过 具名 Slot 来设置当 loading 结束之后需要展示的 UI。

:::demo

```
<template>
  <div style="width: 240px">
    <p>
      <label style="margin-right: 16px;">切换 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>好吃的汉堡</span>
            <div class="bottom card-header">
              <span class="time">{{ currentDate }}</span>
              <el-button type="text" class="button">操作按钮</el-button>
```

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

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

:::

### 渲染多条数据

大多时候, 骨架屏都被用来渲染列表, 当我们需要在从服务器获取数据的时候来渲染一个假的 UI。利用 `count` 这个属性就能控制渲染多少条假的数据在页面上

:::tip 请注意, 请尽可能的将 `count` 的大小保持在最小状态, 即使是假的 UI, DOM 元素多了之后, 照样会引起性能问题, 并且在骨架屏销毁时所消耗的时间也会更长(相对的)。 :::

:::demo

```
<template>
  <div style="width: 400px">
    <p>
      <el-button @click="setLoading">点我重新加载</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>{{ item.name }}</span>
            <div class="bottom card-header">
              <span class="time">{{ currentDate }}</span>
              <el-button type="text" class="button">操作按钮</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: '鹿',
        },
        {
          imgUrl:

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

'https://fuss10.elemecdn.com/0/6f/e35ff375812e6b0020b6b4e8f9583jpeg.jpeg',
          name: '山狮',
        },
      ]
```

```
    },
    methods: {
      setLoading() {
        this.loading = true
        setTimeout(() => (this.loading = false), 2000)
      },
    },
  }
</script>
```

:::

### 防止渲染抖动

有的时候，API 的请求回来的特别快，往往骨架占位刚刚被渲染，真实的数据就已经回来了，用户的界面会突然一闪，此时为了避免这种情况，就需要通过 `throttle` 属性来避免这个问题。

:::demo

```
<template>
  <div style="width: 240px">
    <p>
      <label style="margin-right: 16px;">切换 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>好吃的汉堡</span>
            <div class="bottom card-header">
              <span class="time">{{ currentDate }}</span>
              <el-button type="text" class="button">操作按钮</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

| 参数 | 说明 | 类型 | 可选值 | 默认值 |
| --- | --- | --- | --- | --- |
| animated | 是否使用动画 | boolean | true / false | false |
| count | 渲染多少个 template, 建议使用尽可能小的数字 | number | integer | 1 |
| loading | 是否显示 skeleton 骨架屏 | boolean | true / false | true |
| rows | 骨架屏段落数量 | number | 正整数 | 4 |
| throttle | 延迟占位 DOM 渲染的时间, 单位是毫秒 | number | 正整数 | 0 |

### Skeleton Item Attributes

| 参数 | 说明 | 类型 | 可选值 | 默认值 |
| --- | --- | --- | --- | --- |
| variant | 当前显示的占位元素的样式 | Enum(string) | p / text / h1 / h3 / text / caption / button / image / circle / rect | text |

### Skeleton Slots

| name | description |
| --- | --- |
| default | 用来展示真实 UI |
| template | 用来展示自定义占位符 |