# Table 表格

表格展示数据组。 它们是完全可以自定义的。

表格以一种易于扫描的方式显示信息，以便用户洞察和寻找模型。 表格可以被内嵌在主要内容中，如 卡片（cards）。 它们可以包括：

- 对应的可视化效果
- 导航
- 一个用于查询和操作数据的工具

{{"component": "modules/components/ComponentLinkHeader.js"}}

## 基础表格

一个没有多余装饰的简单例子

{{"demo": "BasicTable.js", "bg": true}}

## 数据表格

`Table` 组件与原生 `<table>` 元素存在密切关联。 这种限制条件导致要构建丰富的数据表格会变得很有挑战性。

The `DataGrid` component is designed for use-cases that are focused on handling large amounts of tabular data. 虽然它的结构相比之下不够灵活，但是有失必有得，牺牲灵活性来换取更强大的功能。 虽然它的结构相比之下不够灵活，但是有失必有得，牺牲灵活性来换取更强大的功能。

{{"demo": "DataTable.js", "bg": "inline"}}

## 紧凑型表格

这是一个简单紧凑型表格，并且没有多余的装饰。

{{"demo": "DenseTable.js", "bg": true}}

## 排序 & 选择

此示例演示了在表格内使用了 选择框组件（`Checkbox`） 以及单击选择行，而且这个表格带有一个自定义的 工具条组件（`Toolbar`） 。 它也展示了如何使用 `TableSortLabel` 组件来给列标题添加样式。

这个表格已被赋予一个固定的宽度，您可以查看如何实现横向滚动。 在表格外部使用 TablePagination 组件，能够防止分页控件的滚动。 (以下的'Custom Table Pagination Action' （自定义表分页操作示例）展示了 TableFooter 中的分页。 )

{{"demo": "EnhancedTable.js", "bg": true}}

## 自定义表格

以下是自定义组件的一个示例。 您可以在 重写文档页面 中了解更多有关此内容的信息。

{{"demo": "CustomizedTables.js", "bg": true}}

### 自定义的分页选项

你也可以使用 `rowsPerPageOptions` 属性来自定义 "Rows per page" 显示的选择项。 你应该提供以下一种数组：

- **数字（numbers）**，而每个数字用作为选择项的标签（label）和值（value）。

```
<TablePagination rowsPerPageOptions={[10, 50]} />
```

- **对象（objects）**，而 `value` 和 `label` 键则相应的对照选择项的标签（label）和值（value）（譬如，当有一个语言字符串为“All” 时你会受益匪浅）。

```
<TablePagination rowsPerPageOptions={[10, 50, { value: -1, label: 'All' }]}
/>
```

### 自定义表格分页操作

`TablePagination` 组件的 `ActionsComponent` 属性能够让您实现一些自定义的行为。

{{"demo": "CustomPaginationActionsTable.js", "bg": true}}

## Sticky header

Here is an example of a table with scrollable rows and fixed column headers. It leverages the `stickyHeader` prop. (⚠ no IE 11 support) It leverages the `stickyHeader` prop. (⚠ no IE 11 support) It leverages the `stickyHeader` prop. (⚠ no IE 11 support)

{{"demo": "StickyHeadTable.js", "bg": true}}

## 按列分组

你可以在一个表头内渲染多个表行来分组列头：

```
<TableHead>
  <TableRow />
  <TableRow />
</TableHead>
```

{{"demo": "ColumnGroupingTable.js", "bg": true}}

## 可折叠的表格

以可扩展行的表格为例，揭示更多信息。 它利用了 `Collapse` 组件。

{{"demo": "CollapsibleTable.js", "bg": true}}

## 跨越表格（Spanning Table）

一个行和列跨越的简单例子。

{{"demo": "SpanningTable.js", "bg": true}}

## 大型列表渲染（Virtualized Table）

以下例子展示了将 react-virtualized 与 `Table` 组件一起使用的方法。 它渲染了 200 多行，并且可以轻松的延展到更多行。 可视化优化了整体的性能。

{{"demo": "ReactVirtualizedTable.js", "bg": true}}

## Unstyled

If you would like to use an unstyled Table, you can use the primitive elements and enhance the table with the unstyled pagination as shown in the demo below.

{{"demo": "TableUnstyled.js"}}

### Customized look and feel

{{"demo": "TableCustomized.js"}}

## Accessibility

(WAI tutorial: https://www.w3.org/WAI/tutorials/tables/)

### Caption 字幕

字幕能够充当表格的表头。 大多数屏幕阅读器能够宣读字幕的内容。 标题可以帮助用户找到表格，了解表格的内容，决定是否要阅读。

{{"demo": "AcccessibleTable.js", "bg": true}}