Overview

Due to the widespread use of `<table>` elements across third-party widgets like calendars and date pickers, Bootstrap's tables are **opt-in**. Add the base class `.table` to any `<table>`, then extend with our optional modifier classes or custom styles. All table styles are not inherited in Bootstrap, meaning any nested tables can be styled independent from the parent.

Using the most basic table markup, here's how `.table`-based tables look in Bootstrap.

{{< table class="table" simplified="false" >}}

Variants

Use contextual classes to color tables, table rows or individual cells.

{{< table.inline >}} {{- range (index $.Site.Data "theme-colors") }} {{- end -}} {{< /table.inline >}}

Class	Heading	Heading
Default	Cell	Cell
{{ .name	title }}	Cell	Cell

{{< highlight html >}}

{{< table.inline >}}
{{- range (index $.Site.Data "theme-colors") }}

...

{{- end -}} {{< /table.inline >}} {{< table.inline >}}
{{- range (index $.Site.Data "theme-colors") }}

... {{- end -}} {{< /table.inline >}} {{< table.inline >}} {{- range (index $.Site.Data "theme-colors") }} ... {{- end -}} {{< /table.inline >}} {{< /highlight >}}
{{< callout info >}} {{< partial "callout-warning-color-assistive-technologies.md" >}} {{< /callout >}}

Accented tables

Striped rows

Use `.table-striped` to add zebra-striping to any table row within the `<tbody>`.

{{< table class="table table-striped" >}}

These classes can also be added to table variants:

{{< table class="table table-dark table-striped" >}}

{{< table class="table table-success table-striped" >}}

Hoverable rows

Add `.table-hover` to enable a hover state on table rows within a `<tbody>`.

{{< table class="table table-hover" >}}

{{< table class="table table-dark table-hover" >}}

These hoverable rows can also be combined with the striped variant:

{{< table class="table table-striped table-hover" >}}

Active tables

Highlight a table row or cell by adding a `.table-active` class.

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

```html
<table class="table">
  <thead>
    ...
  </thead>
  <tbody>
    <tr class="table-active">
      ...
    </tr>
    <tr>
      ...
    </tr>
    <tr>
      <th scope="row">3</th>
      <td colspan="2" class="table-active">Larry the Bird</td>
      <td>@twitter</td>
    </tr>
  </tbody>
</table>
```

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

```html
<table class="table table-dark">
  <thead>
    ...
  </thead>
  <tbody>
    <tr class="table-active">
      ...
    </tr>
```

```
    <tr>
      ...
    </tr>
    <tr>
      <th scope="row">3</th>
      <td colspan="2" class="table-active">Larry the Bird</td>
      <td>@twitter</td>
    </tr>
  </tbody>
</table>
```

How do the variants and accented tables work?

For the accented tables (striped rows, hoverable rows, and active tables), we used some techniques to make these effects work for all our table variants:

- We start by setting the background of a table cell with the `--bs-table-bg` custom property. All table variants then set that custom property to colorize the table cells. This way, we don't get into trouble if semi-transparent colors are used as table backgrounds.
- Then we add an inset box shadow on the table cells with `box-shadow: inset 0 0 0 9999px var(--bs-table-accent-bg);` to layer on top of any specified `background-color`. Because we use a huge spread and no blur, the color will be monotone. Since `--bs-table-accent-bg` is unset by default, we don't have a default box shadow.
- When either `.table-striped`, `.table-hover` or `.table-active` classes are added, the `--bs-table-accent-bg` is set to a semitransparent color to colorize the background.
- For each table variant, we generate a `--bs-table-accent-bg` color with the highest contrast depending on that color. For example, the accent color for `.table-primary` is darker while `.table-dark` has a lighter accent color.
- Text and border colors are generated the same way, and their colors are inherited by default.

Behind the scenes it looks like this:

{{< scss-docs name="table-variant" file="scss/mixins/_table-variants.scss" >}}

Table borders

Bordered tables

Add `.table-bordered` for borders on all sides of the table and cells.

{{< table class="table table-bordered" >}}

[Border color utilities]({{< docsref "/utilities/borders#border-color" >}}) can be added to change colors:

{{< table class="table table-bordered border-primary" >}}

Tables without borders

Add `.table-borderless` for a table without borders.

{{< table class="table table-borderless" >}}

{{< table class="table table-dark table-borderless" >}}

Small tables

Add `.table-sm` to make any `.table` more compact by cutting all cell `padding` in half.

{{< table class="table table-sm" >}}

{{< table class="table table-dark table-sm" >}}

Vertical alignment

Table cells of `<thead>` are always vertical aligned to the bottom. Table cells in `<tbody>` inherit their alignment from `<table>` and are aligned to the the top by default. Use the [vertical align]({{< docsref "/utilities/vertical-align" >}}) classes to re-align where needed.

Heading 1	Heading 2	Heading 3	Heading 4
This cell inherits `vertical-align: middle;` from the table	This cell inherits `vertical-align: middle;` from the table	This cell inherits `vertical-align: middle;` from the table	This here is some placeholder text, intended to take up quite a bit of vertical space, to demonstrate how the vertical alignment works in the preceding cells.
This cell inherits `vertical-align: bottom;` from the table row	This cell inherits `vertical-align: bottom;` from the table row	This cell inherits `vertical-align: bottom;` from the table row	This here is some placeholder text, intended to take up quite a bit of vertical space, to demonstrate how the vertical alignment works in the preceding cells.
This cell inherits `vertical-align: middle;` from the table	This cell inherits `vertical-align: middle;` from the table	This cell is aligned to the top.	This here is some placeholder text, intended to take up quite a bit of vertical space, to demonstrate how the vertical alignment works in the preceding cells.

```
<div class="table-responsive">
  <table class="table align-middle">
    <thead>
      <tr>
        ...
      </tr>
    </thead>
    <tbody>
      <tr>
        ...
      </tr>
      <tr class="align-bottom">
        ...
      </tr>
      <tr>
        <td>...</td>
        <td>...</td>
        <td class="align-top">This cell is aligned to the top.</td>
        <td>...</td>
      </tr>
```

```
      </tbody>
    </table>
  </div>
```

Nesting

Border styles, active styles, and table variants are not inherited by nested tables.

#	First	Last	Handle
1	Mark	Otto	@mdo

Header	Header	Header
A	First	Last
B	First	Last
C	First	Last

#	First	Last	Handle
3	Larry	the Bird	@twitter

```
<table class="table table-striped">
  <thead>
    ...
  </thead>
  <tbody>
    ...
    <tr>
      <td colspan="4">
        <table class="table mb-0">
          ...
        </table>
      </td>
    </tr>
    ...
  </tbody>
</table>
```

How nesting works

To prevent *any* styles from leaking to nested tables, we use the child combinator (`>`) selector in our CSS. Since we need to target all the `td` s and `th` s in the `thead` , `tbody` , and `tfoot` , our selector would look pretty long without it. As such, we use the rather odd looking `.table > :not(caption) > * > *` selector to target all `td` s and `th` s of the `.table` , but none of any potential nested tables.

Note that if you add `<tr>` s as direct children of a table, those `<tr>` will be wrapped in a `<tbody>` by default, thus making our selectors work as intended.

Anatomy

Table head

Similar to tables and dark tables, use the modifier classes `.table-light` or `.table-dark` to make `<thead>` s appear light or dark gray.

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

```
<table class="table">
  <thead class="table-light">
    ...
  </thead>
  <tbody>
    ...
  </tbody>
</table>
```

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

```
<table class="table">
  <thead class="table-dark">
    ...
  </thead>
  <tbody>
    ...
  </tbody>
</table>
```

Table foot

#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter
Footer	Footer	Footer	Footer

```
<table class="table">
  <thead>
    ...
  </thead>
  <tbody>
    ...
  </tbody>
  <tfoot>
    ...
  </tfoot>
</table>
```

Captions

A `<caption>` functions like a heading for a table. It helps users with screen readers to find a table and understand what it's about and decide if they want to read it.

{{< partial "table-content" >}}
 List
  of
users

```
<table class="table table-sm">
  <caption>List of users</caption>
  <thead>
    ...
  </thead>
  <tbody>
    ...
  </tbody>
</table>
```

You can also put the `<caption>` on the top of the table with `.caption-top`.

{{< example >}}

List of users			
#	First	Last	Handle
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

{{< /example >}}

Responsive tables

Responsive tables allow tables to be scrolled horizontally with ease. Make any table responsive across all viewports by wrapping a `.table` with `.table-responsive`. Or, pick a maximum breakpoint with which to have a responsive table up to by using `.table-responsive{-sm|-md|-lg|-xl|-xxl}`.

{{< callout warning >}}

Vertical clipping/truncation

Responsive tables make use of `overflow-y: hidden`, which clips off any content that goes beyond the bottom or top edges of the table. In particular, this can clip off dropdown menus and other third-party widgets. {{< /callout >}}

Always responsive

Across every breakpoint, use `.table-responsive` for horizontally scrolling tables.

#	Heading	Heading	Heading	Heading	Heading	Heading	Heading	Heading
1	Cell	Cell	Cell	Cell	Cell	Cell	Cell	Cell
2	Cell	Cell	Cell	Cell	Cell	Cell	Cell	Cell
3	Cell	Cell	Cell	Cell	Cell	Cell	Cell	Cell

```html
<div class="table-responsive">
  <table class="table">
    ...
  </table>
</div>
```

Breakpoint specific

Use `.table-responsive{-sm|-md|-lg|-xl|-xxl}` as needed to create responsive tables up to a particular breakpoint. From that breakpoint and up, the table will behave normally and not scroll horizontally.

These tables may appear broken until their responsive styles apply at specific viewport widths.

{{< tables.inline >}} {{ range $.Site.Data.breakpoints }} {{ if not (eq . "xs") }}

#	Heading	Heading	Heading	Heading	Heading	Heading	Heading	Heading
1	Cell	Cell	Cell	Cell	Cell	Cell	Cell	Cell
2	Cell	Cell	Cell	Cell	Cell	Cell	Cell	Cell
3	Cell	Cell	Cell	Cell	Cell	Cell	Cell	Cell

{{ end -}} {{- end -}} {{< /tables.inline >}}

{{< highlight html >}} {{< tables.inline >}} {{- range $.Site.Data.breakpoints -}} {{- if not (eq . "xs") }}

…

{{ end -}} {{- end -}} {{< /tables.inline >}} {{< /highlight >}}

Sass

Variables

{{< scss-docs name="table-variables" file="scss/_variables.scss" >}}

Loop

{{< scss-docs name="table-loop" file="scss/_variables.scss" >}}

Customizing

- The factor variables (`$table-striped-bg-factor` , `$table-active-bg-factor` & `$table-hover-bg-factor`) are used to determine the contrast in table variants.
- Apart from the light & dark table variants, theme colors are lightened by the `$table-bg-level` variable.