

Grid system

Bootstrap’s grid system uses a series of containers, rows, and columns to layout and align content. It’s built with [flexbox](#) and is fully responsive. Below is an example and an in-depth look at how the grid comes together.

New to or unfamiliar with flexbox? [Read this CSS Tricks flexbox guide](#) for background, terminology, guidelines, and code snippets.

Container

Containers provide a means to center and horizontally pad your site’s contents. Use `Container` for a responsive pixel width.

1 of 1

```
<Container>
  <Row>
    <Col>1 of 1</Col>
  </Row>
</Container>
```

Fluid Container

You can use `<Container fluid />` for width: 100% across all viewport and device sizes.

1 of 1

```
<Container fluid>
  <Row>
    <Col>1 of 1</Col>
  </Row>
</Container>
```

You can set breakpoints for the `fluid` prop. Setting it to a breakpoint (`sm`, `md`, `lg`, `xl`) will set the `Container` as fluid until the specified breakpoint.

1 of 1

```
<Container fluid="md">
  <Row>
    <Col>1 of 1</Col>
  </Row>
</Container>
```

Auto-layout columns

When no column widths are specified the `Col` component will render equal width columns

1 of 2

2 of 2

1 of 3

2 of 3

3 of 3

```
<Container>
  <Row>
    <Col>1 of 2</Col>
    <Col>2 of 2</Col>
  </Row>
  <Row>
    <Col>1 of 3</Col>
    <Col>2 of 3</Col>
    <Col>3 of 3</Col>
  </Row>
</Container>
```

Setting one column width

Auto-layout for flexbox grid columns also means you can set the width of one column and have the sibling columns automatically resize around it. You may use predefined grid classes (as shown below), grid mixins, or inline widths. Note that the other columns will resize no matter the width of the center column.

| | | |
|--------|----------------|--------|
| 1 of 3 | 2 of 3 (wider) | 3 of 3 |
|--------|----------------|--------|

| | | |
|--------|----------------|--------|
| 1 of 3 | 2 of 3 (wider) | 3 of 3 |
|--------|----------------|--------|

```
<Container>
  <Row>
    <Col>1 of 3</Col>
    <Col xs={6}>2 of 3 (wider)</Col>
    <Col>3 of 3</Col>
  </Row>
  <Row>
    <Col>1 of 3</Col>
    <Col xs={5}>2 of 3 (wider)</Col>
    <Col>3 of 3</Col>
  </Row>
</Container>
```

Variable width content

Set the column value (for any breakpoint size) to **"auto"** to size columns based on the natural width of their content.

| | | |
|--------|------------------------|--------|
| 1 of 3 | Variable width content | 3 of 3 |
|--------|------------------------|--------|

| | | |
|--------|------------------------|--------|
| 1 of 3 | Variable width content | 3 of 3 |
|--------|------------------------|--------|

```
<Container>
  <Row className="justify-content-md-center">
    <Col xs lg="2">
      1 of 3
    </Col>
    <Col md="auto">Variable width content</Col>
    <Col xs lg="2">
      3 of 3
    </Col>
  </Row>
  <Row>
    <Col>1 of 3</Col>
    <Col md="auto">Variable width content</Col>
    <Col xs lg="2">
      3 of 3
    </Col>
  </Row>
</Container>
```

Responsive grids

The **Col** lets you specify column widths across 5 breakpoint sizes (xs, sm, md, lg, and xl). For every breakpoint, you can specify the amount of columns to span, or set the prop to **<Col lg={true} />** for auto layout widths.

| | |
|------|------|
| sm=8 | sm=4 |
|------|------|

| | | |
|---------|---------|---------|
| sm=true | sm=true | sm=true |
|---------|---------|---------|

```
<Container>
  <Row>
    <Col sm={8}>sm=8</Col>
    <Col sm={4}>sm=4</Col>
  </Row>
  <Row>
    <Col sm>sm=true</Col>
    <Col sm>sm=true</Col>
    <Col sm>sm=true</Col>
  </Row>
</Container>
```

You can also mix and match breakpoints to create different grids depending on the screen size.

| | |
|--|--|
| | |
|--|--|

| | | |
|------------|-----------|-----------|
| xs=12 md=8 | | xs=6 md=4 |
| | | |
| xs=6 md=4 | xs=6 md=4 | xs=6 md=4 |
| | | |
| xs=6 | xs=6 | |

```
<Container>
  { /* Stack the columns on mobile by making one full-width and the other half-width */ }
  <Row>
    <Col xs={12} md={8}>
      xs=12 md=8
    </Col>
    <Col xs={6} md={4}>
      xs=6 md=4
    </Col>
  </Row>

  { /* Columns start at 50% wide on mobile and bump up to 33.3% wide on desktop */ }
  <Row>
    <Col xs={6} md={4}>
      xs=6 md=4
    </Col>
    <Col xs={6} md={4}>
      xs=6 md=4
    </Col>
    <Col xs={6} md={4}>
      xs=6 md=4
    </Col>
  </Row>

  { /* Columns are always 50% wide, on mobile and desktop */ }
  <Row>
    <Col xs={6}>xs=6</Col>
    <Col xs={6}>xs=6</Col>
  </Row>
</Container>
```

The `Col` breakpoint props also have a more complicated `object` prop form: `{span: number, order: number, offset: number}` for specifying offsets and ordering effects.

You can use the `order` property to control the **visual order** of your content.

| | | |
|----------------------|-------------------|------------------|
| First, but unordered | Third, but second | Second, but last |
|----------------------|-------------------|------------------|

```
<Container>
  <Row>
    <Col xs>First, but unordered</Col>
    <Col xs={{ order: 12 }}>Second, but last</Col>
    <Col xs={{ order: 1 }}>Third, but second</Col>
  </Row>
</Container>
```

The `order` property also supports `first` (`order: -1`) and `last` (`order: $columns+1`).

| | | |
|------------------|-----------------------|-----------------|
| Third, but first | Second, but unordered | First, but last |
|------------------|-----------------------|-----------------|

```
<Container>
  <Row>
    <Col xs={{ order: 'last' }}>First, but last</Col>
    <Col xs>Second, but unordered</Col>
    <Col xs={{ order: 'first' }}>Third, but first</Col>
  </Row>
</Container>
```

For offsetting grid columns you can set an `offset` value or for a more general layout, use the margin class utilities.

md=4

md={{ span: 4, offset: 4 }}

md={{ span: 3, offset: 3 }}

md={{ span: 3, offset: 3 }}

md={{ span: 6, offset: 3 }}

```
<Container>
  <Row>
    <Col md={4}>md=4</Col>
    <Col md={{ span: 4, offset: 4 }}>`md={{ span: 4, offset: 4 }}`</Col>
  </Row>
  <Row>
    <Col md={{ span: 3, offset: 3 }}>`md={{ span: 3, offset: 3 }}`</Col>
    <Col md={{ span: 3, offset: 3 }}>`md={{ span: 3, offset: 3 }}`</Col>
  </Row>
  <Row>
    <Col md={{ span: 6, offset: 3 }}>`md={{ span: 6, offset: 3 }}`</Col>
  </Row>
</Container>
```

Setting column widths in Row

The `Row` lets you specify column widths across 5 breakpoint sizes (xs, sm, md, lg, and xl). For every breakpoint, you can specify the amount of columns that will fit next to each other.

| | |
|--------|--------|
| 1 of 2 | 2 of 2 |
| 1 of 3 | 2 of 3 |
| 3 of 3 | |

```
<Container>
  <Row xs={2} md={4} lg={6}>
    <Col>1 of 2</Col>
    <Col>2 of 2</Col>
  </Row>
  <Row xs={1} md={2}>
    <Col>1 of 3</Col>
    <Col>2 of 3</Col>
    <Col>3 of 3</Col>
  </Row>
</Container>
```

Note that `Row` column widths will override `Col` widths set on lower breakpoints when viewed on larger screens. The `<Col xs={6} />` size will be overridden by `<Row md={4} />` on medium and larger screens.

| | | |
|--------|--------|--------|
| 1 of 3 | 2 of 3 | 3 of 3 |
|--------|--------|--------|

```
<Container>
  <Row md={4}>
    <Col>1 of 3</Col>
    <Col xs={6}>2 of 3</Col>
    <Col>3 of 3</Col>
  </Row>
</Container>
```

API

Container

```
import Container from 'react-bootstrap/Container' 
```

| Name | Type | Default | Description |
|----------|---|--------------------------|--|
| as | elementType | <code><div></code> | You can use a custom element for this component |
| fluid | <code>true</code> <code>"sm"</code> <code>"md"</code> <code>"lg"</code> <code>"xl"</code> | <code>false</code> | Allow the Container to fill all of its available horizontal space. |
| bsPrefix | string | <code>'container'</code> | Change the underlying component CSS base class name and modifier class names prefix. This is an escape hatch for working with heavily customized bootstrap css. |

Row

```
import Row from 'react-bootstrap/Row' 
```

| Name | Type | Default | Description |
|-----------|---------------------------------|--------------------------|--|
| as | elementType | <code><div></code> | You can use a custom element type for this component. |
| lg | number { cols: number } | | The number of columns that will fit next to each other on large devices (≥992px) |
| md | number { cols: number } | | The number of columns that will fit next to each other on medium devices (≥768px) |
| noGutters | boolean | <code>false</code> | Removes the gutter spacing between <code>Col</code> s as well as any added negative margins. |
| sm | number { cols: number } | | The number of columns that will fit next to each other on small devices (≥576px) |
| xl | number { cols: number } | | The number of columns that will fit next to each other on extra large devices (≥1200px) |
| xs | number { cols: number } | | The number of columns that will fit next to each other on extra small devices (<576px) |
| bsPrefix | string | <code>'row'</code> | Change the underlying component CSS base class name and modifier class names prefix. This is an escape hatch for working with heavily customized bootstrap css. |

Col

```
import Col from 'react-bootstrap/Col' 
```

| Name | Type | Default | Description |
|----------|--|--------------------------|--|
| as | elementType | <code><div></code> | You can use a custom element type for this component. |
| lg | boolean "auto" number { span: boolean "auto" number, offset: number, order: "first" "last" number } | | The number of columns to span on large devices (≥992px) |
| md | boolean "auto" number { span: boolean "auto" number, offset: number, order: "first" "last" number } | | The number of columns to span on medium devices (≥768px) |
| sm | boolean "auto" number { span: boolean "auto" number, offset: number, order: "first" "last" number } | | The number of columns to span on small devices (≥576px) |
| xl | boolean "auto" number { span: boolean "auto" number, offset: number, order: "first" "last" number } | | The number of columns to span on extra large devices (≥1200px) |
| xs | boolean "auto" number { span: boolean "auto" number, offset: number, order: "first" "last" number } | | The number of columns to span on extra small devices (<576px) |
| bsPrefix | string | <code>'col'</code> | Change the underlying component CSS base class name and modifier class names prefix. This is an escape hatch for working with heavily customized bootstrap css. |

Search...

[Getting started](#)

[Layout](#)

[Grid](#)

[Media](#)

[Components](#)

[Utilities](#)

[Migrating](#)

[About](#)

