Search...

Getting started
Components

M4Q
About
Population
Constructor
Ajax
Animation
Loops
Visibility
Effects
Subtree functions
Attributes
Html, text and value
Css and classes
Position and size
Manipulation
DataSet
Events
Utils

Base
Containers
Grid system
Typography
Tables
Forms
Buttons
Images
Figures
Lists

Form components
Checkbox
File input
Input
Input material
Keypad
Rating
Radio
Select
Slider
Double Slider
Spinner
Switch
Tag input
Textarea

Menus
App bar
Bottom navigation
Bottom sheet
Menu
Ribbon Menu
Side bar
Side navigation

Controls
Accordion
Badge
Carousel
Cards
Cube
Counter
Charms
Chat
Donut
Image compare
Image magnifier
Gravatar
List
ListView
Master
NavView
Panels
Progress & Activity
Streamer
Stepper
Splitter
Tabs
Tabs material
Table
Tiles
TreeView
Wizard

Information
Dialogs
Info box
Hints
Notify system
Popovers
Toasts
Windows

Date & time
Calendar
Calendar picker
Date picker
Time picker

Grid system

Use our powerful mobile-first flexbox grid to build layouts of all shapes and sizes thanks to a twelve column system, five default responsive tiers, Less variables and mixins, and dozens of predefined classes.

About

Metro 4 grid system uses a series of containers, rows, and columns to layout and align content.

1	2	3	4	5	6	7	8	9	10	11	12
cell-3			cell-3			cell-3			cell-3		
				cell-4 + offset-4							
cell-3						cell-6 + offset-3					
				cell-4 + offset-4				1	1	1	1
nested			nested			nested			nested		

```
<div class="grid">
    <div class="row">
        <div class="cell"><div>1</div></div>
        <div class="cell"><div>2</div></div>
        <div class="cell"><div>3</div></div>
        <div class="cell"><div>4</div></div>
        <div class="cell"><div>5</div></div>
        <div class="cell"><div>6</div></div>
        <div class="cell"><div>7</div></div>
        <div class="cell"><div>8</div></div>
        <div class="cell"><div>9</div></div>
        <div class="cell"><div>10</div></div>
        <div class="cell"><div>11</div></div>
        <div class="cell"><div>12</div></div>
    </div>
    <div class="row">
        <div class="cell-3"><div>cell-3</div></div>
        <div class="cell-3"><div>cell-3</div></div>
        <div class="cell-3"><div>cell-3</div></div>
        <div class="cell-3"><div>cell-3</div></div>
    </div>
    <div class="row">
        <div class="cell-4 offset-4"><div>cell-4 + offset-4</div></div>
    </div>
    <div class="row">
        <div class="cell-3"><div>cell-3</div></div>
        <div class="cell-6 offset-3"><div>cell-6 + offset-3</div></div>
    </div>
    <div class="row">
        <div class="cell-4 offset-4"><div>cell-4 + offset-4</div></div>
        <div class="cell"><div>1</div></div>
        <div class="cell"><div>1</div></div>
        <div class="cell"><div>1</div></div>
        <div class="cell"><div>1</div></div>
    </div>
    <div class="row">
        <div class="cell">
            <div class="row">
                <div class="cell"><div>nested</div></div>
                <div class="cell"><div>nested</div></div>
                <div class="cell"><div>nested</div></div>
                <div class="cell"><div>nested</div></div>
            </div>
        </div>
    </div>
</div>
```

How it works

Metro 4 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.

One of three columns	One of three columns	One of three columns

The above example creates three equal-width columns on small, medium, large, and extra large devices using our predefined grid classes. Those columns are centered in the page with the parent `.container`.

Breaking it down, here's how it works:

- Containers provide a means to center your site's contents. Use `.container` for fixed width or `.container-fluid` for full width.
- User rows grouping with `.grid` container
- Rows are horizontal groups of columns that ensure your columns are lined up properly. We use the negative margin method on `.row` to ensure all your content is aligned properly down the left side.
- Content should be placed within columns, and only columns may be immediate children of rows.

Table of contents

- Grid columns without a set width will automatically layout with equal widths. For example, four instances of `.cell-sm` will each automatically be `25%` wide for small breakpoints.
- Column classes indicate the number of columns you'd like to use out of the possible 12 per row. So, if you want three equal-width columns, you can use `.cell-sm-4` .
- Column widths are set in percentallyes, so they're always fluid and sized relative to their parent element.
- Columns have horizontal padding to create the gutters between individual columns, however, you can remove the margin from rows and padding from columns with `.no-gap` on the `.row` or on the `.grid` for all rows in grid.
- There are five grid tiers, one for each responsive breakpoint: small (sm), medium (md), large (lg), extra large (xl) and ultra large (xxl).
- Grid tiers are based on minimum widths, meaning they apply to that one tier and all those above it (e.g., `.cell-md-4` applies to medium, large, extra large and ultra large devices).
- You can use predefined grid classes for more semantic markup.

```
<!-- Grouping rows -->
<div class="grid">
    <div class="row">
        <div class="cell"></div>
        ...
        <div class="cell"></div>
    </div>
    ...
    <div class="row">
        <div class="cell"></div>
        ...
        <div class="cell"></div>
    </div>
</div>

<!-- Single row -->
<div class="row">
    <div class="cell"></div>
    ...
    <div class="cell"></div>
</div>

<!-- Nested row -->
<div class="row">
    <div class="cell">
        <div class="row">
            <div class="cell"></div>
            ...
            <div class="cell"></div>
        </div>
    </div>
</div>
```

Colspan

If you need to combine columns, there are appropriate classes for this: `.cell-1` through `.cell-12` and specific media offsets `.cell-*-1` through `.cell-*-12` .

<table>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td colspan="2">colspan2</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td colspan="3">colspan3</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td colspan="4">colspan4</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
</table>

```
<div class="row">
    <div class="cell-4"></div>
    <div class="cell-4"></div>
    <div class="cell-4"></div>
</div>
<div class="row">
    <div class="colspan-4"></div>
    <div class="colspan-4"></div>
    <div class="colspan-4"></div>
</div>
```

Offset

If you need to shift the column, there are appropriate classes for this: `.offset-1` through `.offset-12` and specific media offsets `.offset-*-1` through `.offset-*-12` .

<table>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td></td><td colspan="11">offset1</td></tr>
<tr><td></td><td></td><td colspan="10">offset2</td></tr>
<tr><td></td><td></td><td></td><td colspan="9">offset3</td></tr>
</table>

```
<div class="row">
    <div class="cell offset-1"></div>
</div>
<div class="row">
    <div class="cell offset-2"></div>
</div>
<div class="row">
    <div class="cell offset-3"></div>
</div>
```

Media columns

There are five grid tiers, one for each responsive breakpoint: small (sm), medium (md), large (lg), extra large (xl) and ultra large (xxl).

Breakpoints	min-width
sm	576px
md	768px
lg	992px
xl	1200px
xxl	1452px

You can use predefined grid classes for more semantic markup.

Target	Media	Example
colspan	cell-@-*	cell-md-3
offset	offset-@-*	offset-md-3
ordering	order-@-*	order-md-3
width auto	cell-@-auto	cell-md-auto
width 100%	cell-@-full	cell-md-full
width 100%	cell-@	cell-md
width 25%	cell-@-quarter	cell-md-quarter
width 50%	cell-@-half	cell-md-half
width 1/3	cell-@-one-third	cell-md-one-third
width 2/3	cell-@-two-third	cell-md-two-third

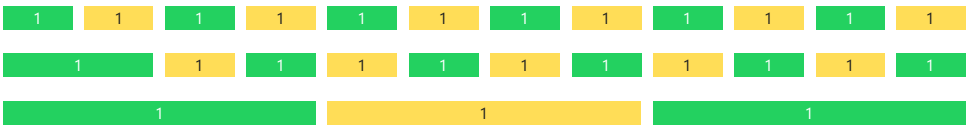
1	1	1	1

```
<div class="row">
    <div class="cell-sm-full cell-md-one-third cell-lg-3"></div>
    <div class="cell-sm-full cell-md-two-third cell-lg-3"></div>
    <div class="cell-sm-full cell-md-half cell-lg-3"></div>
    <div class="cell-sm-full cell-md-half cell-lg-3"></div>
</div>
```

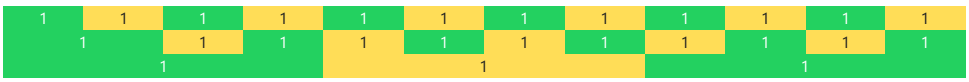
Gaps

The gaps between columns in our predefined grid classes can be removed with `.no-gap`. This removes the negative margins from `.row` and the horizontal padding from all immediate children columns.

With gaps



No gaps



```
<div class="grid no-gap">...</div>
<div class="row no-gap">...</div>
```

Stub column

With class `.stub` you can create column with fixed width.

Stub column	1

```
<div class="row">
    <div class="stub" style="width: 300px"></div>
    <div class="cell"></div>
</div>
```

Ordering

You can use classes `.order-*` to set order columns.

Order 1	Order 2

```
<div class="row">
    <div class="cell order-2">Order 2</div>
    <div class="cell order-1">Order 1</div>
</div>
```

Nested rows

Metro 4 supports nested rows.

Cell 1	Cell 2	Cell 2	Cell 3
Nested row	Nested row / Nested row / Nested row / Nested row / Nested row / Nested row		Nested row
Nested row	Nested row		Nested row
Nested row	Nested row		Nested row
Nested row	Nested row		Nested row

```
<div class="row">
    <div class="cell">
        <div class="row">
            <div class="cell"></div>
            ...
            <div class="cell"></div>
        </div>
    </div>
</div>
```

Alignment columns

To vertical and horizontal align column you can use flexbox alignment utilities classes.

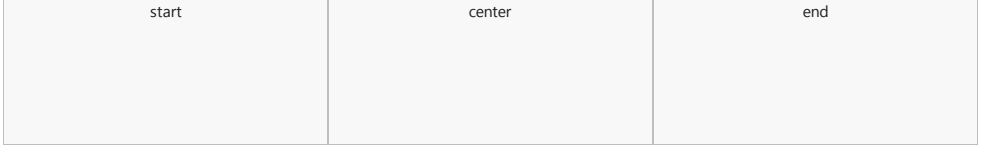
Vertical alignment

For vertical alignment all cells in a row: `.flex-align-start`, `.flex-align-end`, `.flex-align-center`, `.flex-align-stretch`, `.flex-align-baseline`,

For vertical alignment self cell in a row: `.flex-align-self-start`, `.flex-align-self-end`, `.flex-align-self-center`, `.flex-align-self-stretch`, `.flex-align-self-baseline`,

cell	cell	cell
cell	cell	cell
cell	cell	cell

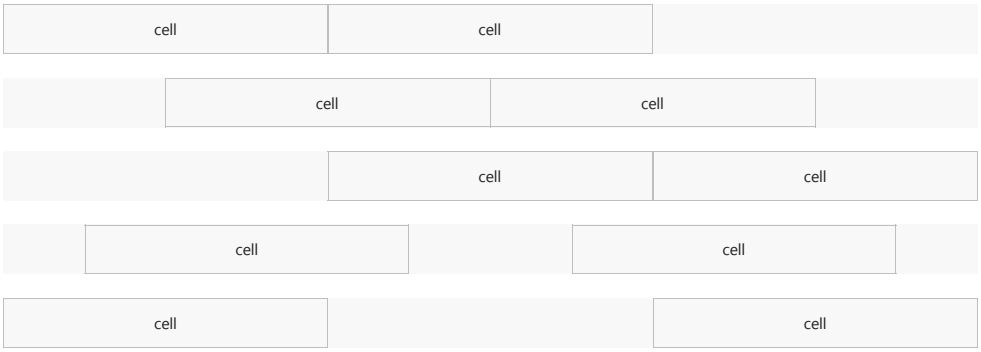
```
<div class="row flex-align-start">
    <div class="cell"></div>
    <div class="cell"></div>
    <div class="cell"></div>
</div>
<br/>
<div class="row flex-align-center">
    <div class="cell"></div>
    <div class="cell"></div>
    <div class="cell"></div>
</div>
<br />
<div class="row flex-align-end">
    <div class="cell"></div>
    <div class="cell"></div>
    <div class="cell"></div>
</div>
```




```
<div class="row">
    <div class="cell flex-align-self-start"></div>
    <div class="cell flex-align-self-center"></div>
    <div class="cell flex-align-self-end"></div>
</div>
```

Horizontal alignment

For horizontal alignment cells in a row: `.flex-justify-start` , `.flex-justify-end` , `.flex-justify-center` , `.flex-justify-between` , `.flex-justify-around` ,



Also you can use media alignment classes such as `.flex-align-start-*` , `.flex-justify-start-*` where asterisk is responsive breakpoint (sm, md, lg, xl, xxl)