BULMA

Modern CSS Framework

Overview

Getting started

You only need 1 CSS file to use Bulma

There are several ways to get started with Bulma. You can either:

- use npm to install the Bulma package
- use the cdnjs CDN to link to the Bulma stylesheet
- use the GitHub repository to get the latest development version

Getting started

1. **Use NPM (recommended):**
   ```
   npm install bulma
   ```
2. **Use the cdnjs CDN**
   https://cdnjs.com/libraries/bulma
3. **Download from the repository**
   https://github.com/jgthms/bulma/tree/master/css
4. If you want to use icons with Bulma, don't forget to include Font Awesome:

   ```
   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">
   ```

Code requirements

1. Use the HTML5 doctype

```
<!DOCTYPE html>
```

2. Add the responsive viewport meta tag

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <title>Hello Bulma!</title>
        <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">
        <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.6.1/css/bulma.min.css">
    </head>
    <body>
        <section class="section">
            <div class="container">
                <h1 class="title">Hello World</h1>
                <p class="subtitle">My first website with <strong>Bulma</strong>!</p>
            </div>
        </section>
    </body>
</html>
```

Starter template

Customizing with Sass

Create your own theme with a simple set of variables

1. Download the source files:

   ```
   npm install bulma
   ```

   or clone the repository:

   https://github.com/jgthms/bulma

2. Set your variables
3. See the result: before and after

Classes

- Bulma is simply a collection of CSS classes. Write the HTML code you want.
- Bulma is a CSS framework, meaning that the end result is simply a single `.css` file: https://github.com/jgthms/bulma/blob/master/css/bulma.css
- Because Bulma solely comprises CSS classes, the HTML code you write has no impact on the styling of your page.
- That's why `.input` exists as a class, so you can choose which `<input type="text">` elements you want to style.

Modular

Just import what you **need**

- Bulma consists of `39` `.sass` files that you can import individually.
- For example, let's say you only want the Bulma columns.
- The file is located in the `bulma/sass/grid` folder.
- Simply import the utilities dependencies, and then the files you need directly:

```
@import "bulma/sass/utilities/_all"
@import "bulma/sass/grid/columns"
```

Responsiveness

Bulma is a **mobile-first** framework

Vertical by default

Every element in Bulma is mobile-first and optimizes for vertical reading, so by default on mobile:

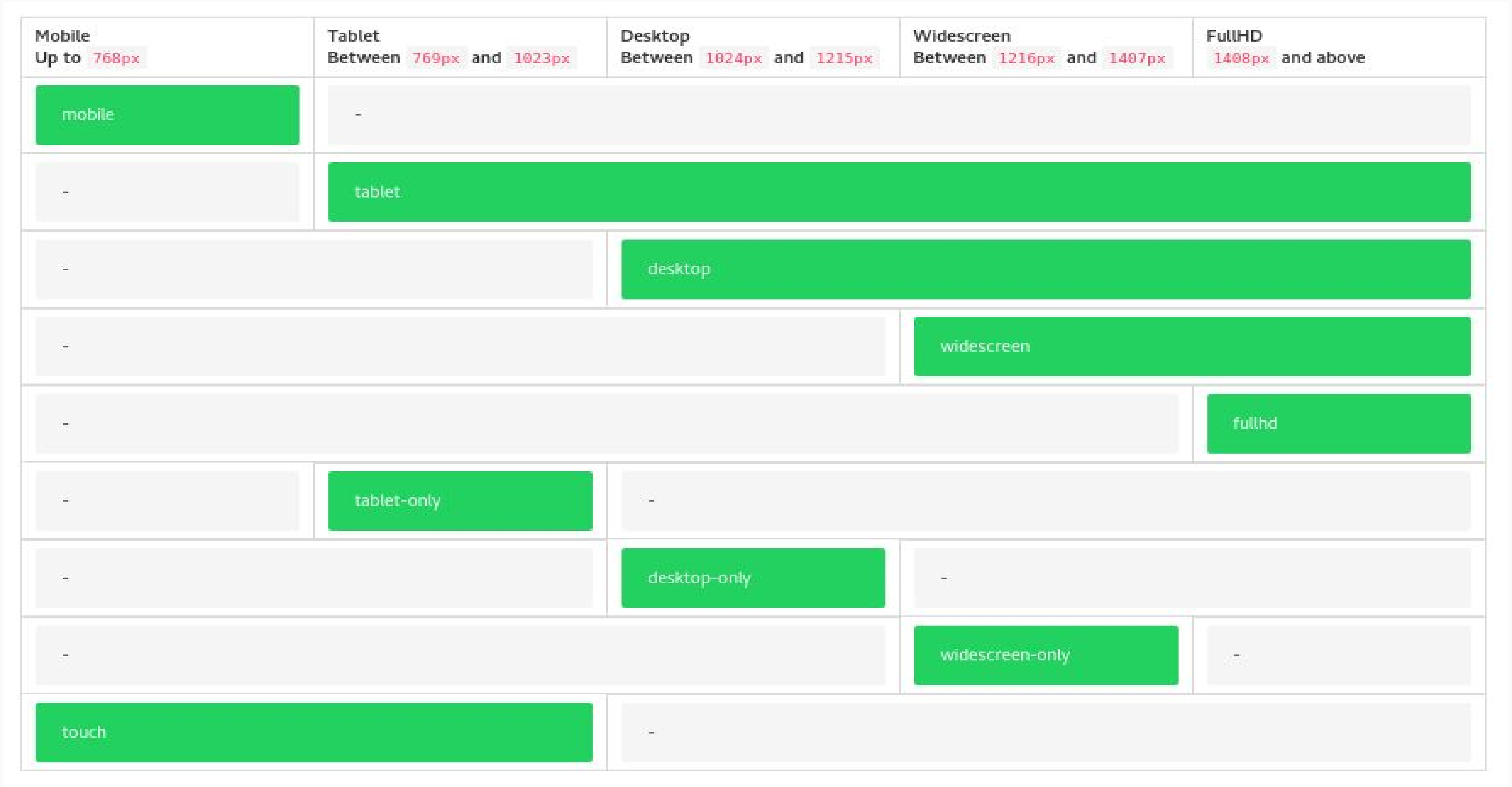
- columns are stacked vertically
- the level component will show its children stacked vertically
- the nav menu will be hidden

For example, you can enforce the horizontal layout for both columns or nav by appending the `is-mobile` modifier.

Breakpoints

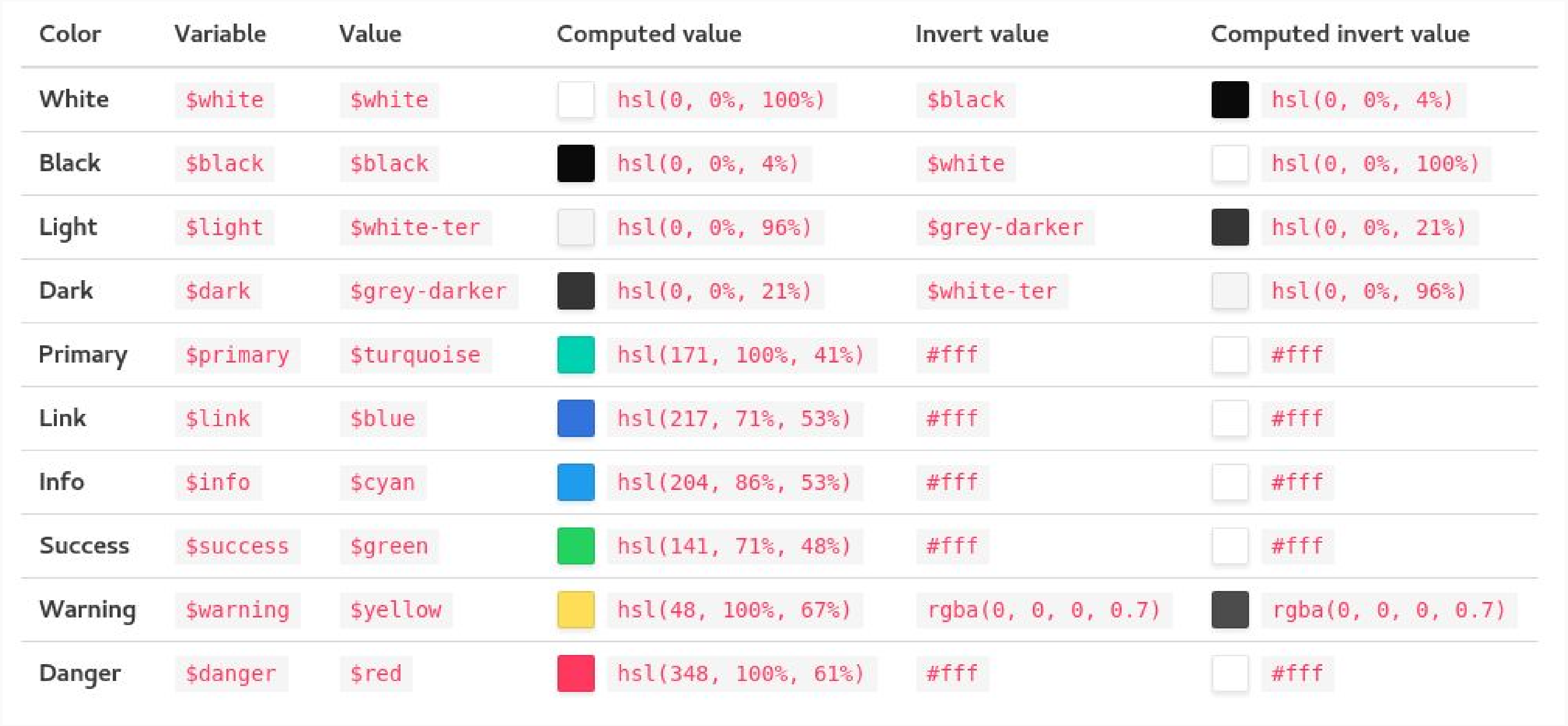
Bulma has 5 breakpoints:

- mobile: up to 768px
- tablet: from 769px
- desktop: from 1024px
- widescreen: from 1216px
- New! fullhd: from 1408px



Mobile Up to `768px`	Tablet Between `769px` and `1023px`	Desktop Between `1024px` and `1215px`	Widescreen Between `1216px` and `1407px`	FullHD `1408px` and above
mobile	-			
-	tablet			
-		desktop		
-			widescreen	
-				fullhd
-	tablet-only	-		
-		desktop-only	-	
-			widescreen-only	-
touch		-		

Breakpoints



Color	Variable	Value	Computed value	Invert value	Computed invert value
White	`$white`	`$white`	⬜ `hsl(0, 0%, 100%)`	`$black`	⬛ `hsl(0, 0%, 4%)`
Black	`$black`	`$black`	⬛ `hsl(0, 0%, 4%)`	`$white`	⬜ `hsl(0, 0%, 100%)`
Light	`$light`	`$white-ter`	⬜ `hsl(0, 0%, 96%)`	`$grey-darker`	⬛ `hsl(0, 0%, 21%)`
Dark	`$dark`	`$grey-darker`	⬛ `hsl(0, 0%, 21%)`	`$white-ter`	⬜ `hsl(0, 0%, 96%)`
Primary	`$primary`	`$turquoise`	🟩 `hsl(171, 100%, 41%)`	`#fff`	⬜ `#fff`
Link	`$link`	`$blue`	🟦 `hsl(217, 71%, 53%)`	`#fff`	⬜ `#fff`
Info	`$info`	`$cyan`	🟦 `hsl(204, 86%, 53%)`	`#fff`	⬜ `#fff`
Success	`$success`	`$green`	🟩 `hsl(141, 71%, 48%)`	`#fff`	⬜ `#fff`
Warning	`$warning`	`$yellow`	🟨 `hsl(48, 100%, 67%)`	`rgba(0, 0, 0, 0.7)`	⬛ `rgba(0, 0, 0, 0.7)`
Danger	`$danger`	`$red`	🟥 `hsl(348, 100%, 61%)`	`#fff`	⬜ `#fff`

Colors

Functions

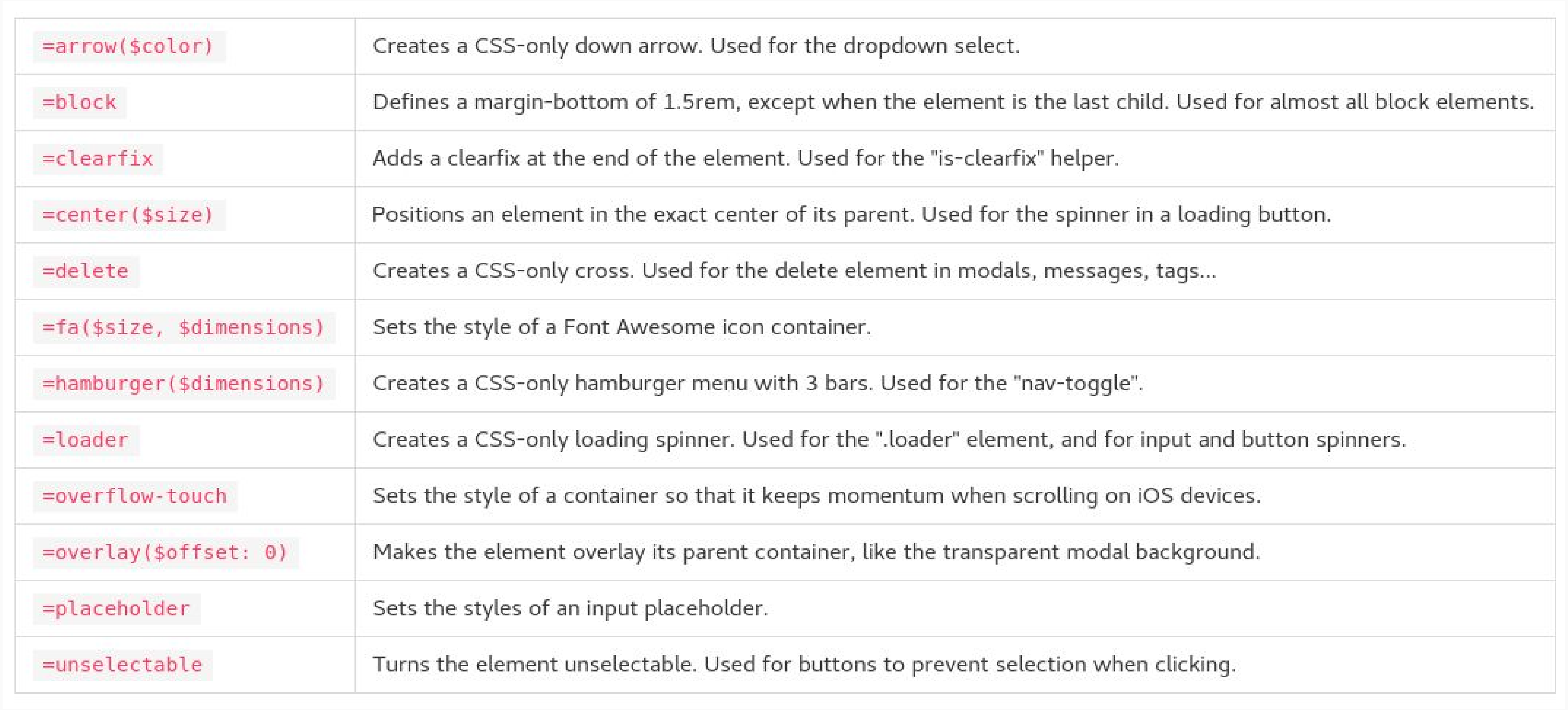
Utility functions to calculate colors and other values

Bulma uses 3 custom functions to help define the values and colors dynamically:

- `powerNumber($number, $exp):` calculates the value of a number exposed to another one. Returns a number.
- `colorLuminance($color):` defines if a color is dark or light. Return a decimal number between 0 and 1 where <= 0.5 is dark and > 0.5 is light.
- `findColorInvert($color):` returns either 70% transparent black or 100% opaque white depending on the luminance of the color.

Mixins

Utility mixins for custom elements and responsive helpers

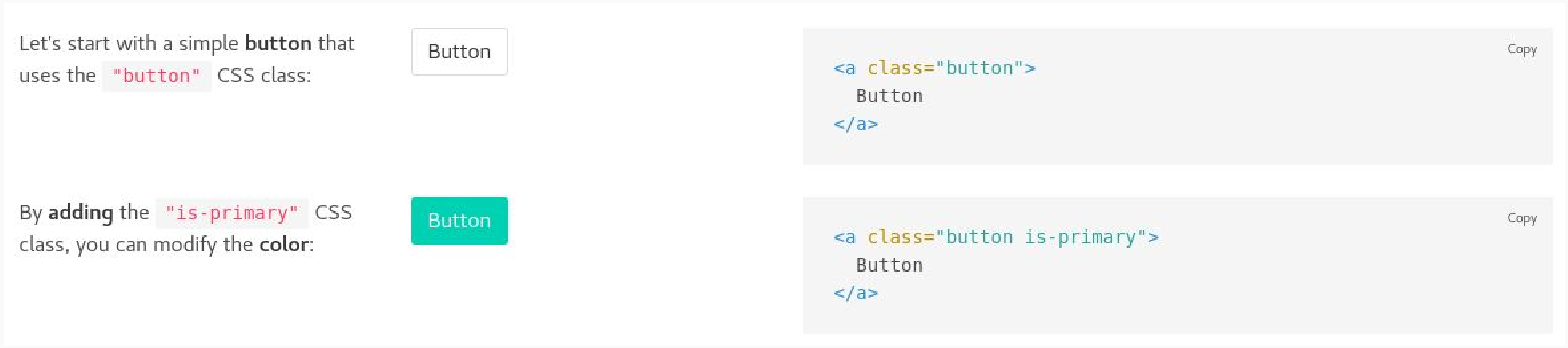


`=arrow($color)`	Creates a CSS-only down arrow. Used for the dropdown select.
`=block`	Defines a margin-bottom of 1.5rem, except when the element is the last child. Used for almost all block elements.
`=clearfix`	Adds a clearfix at the end of the element. Used for the "is-clearfix" helper.
`=center($size)`	Positions an element in the exact center of its parent. Used for the spinner in a loading button.
`=delete`	Creates a CSS-only cross. Used for the delete element in modals, messages, tags...
`=fa($size, $dimensions)`	Sets the style of a Font Awesome icon container.
`=hamburger($dimensions)`	Creates a CSS-only hamburger menu with 3 bars. Used for the "nav-toggle".
`=loader`	Creates a CSS-only loading spinner. Used for the ".loader" element, and for input and button spinners.
`=overflow-touch`	Sets the style of a container so that it keeps momentum when scrolling on iOS devices.
`=overlay($offset: 0)`	Makes the element overlay its parent container, like the transparent modal background.
`=placeholder`	Sets the styles of an input placeholder.
`=unselectable`	Turns the element unselectable. Used for buttons to prevent selection when clicking.

Modifiers

Modifiers syntax

Most Bulma elements have alternative styles. To apply them, you only need to append one of the modifier classes. They all start with is- or has-.



Let's start with a simple **button** that uses the `"button"` CSS class:

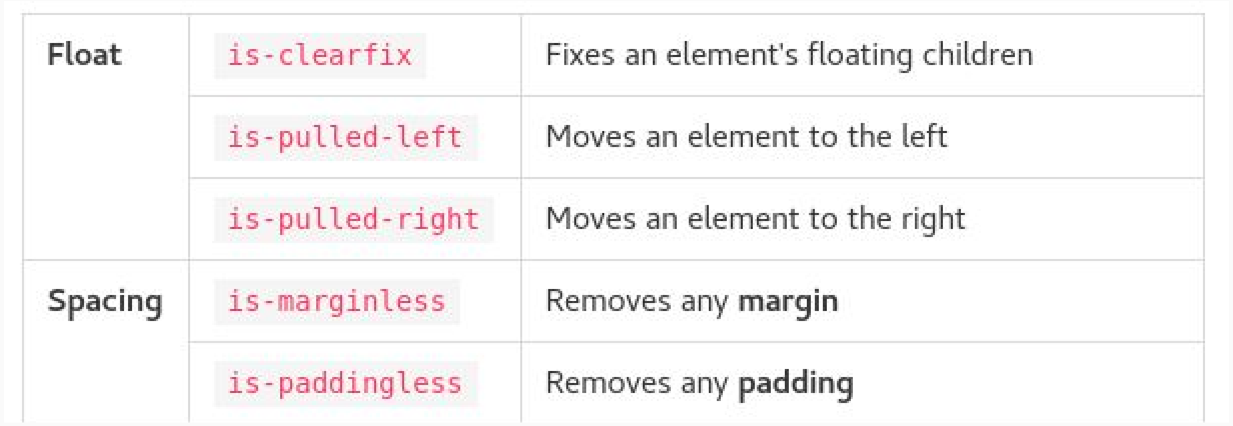
```
<a class="button">
  Button
</a>
```

By **adding** the `"is-primary"` CSS class, you can modify the **color**:

```
<a class="button is-primary">
  Button
</a>
```

Helpers

You can apply responsive helper classes to almost any element, in order to alter its style based upon the browser's width.



Float	`is-clearfix`	Fixes an element's floating children
	`is-pulled-left`	Moves an element to the left
	`is-pulled-right`	Moves an element to the right
Spacing	`is-marginless`	Removes any **margin**
	`is-paddingless`	Removes any **padding**

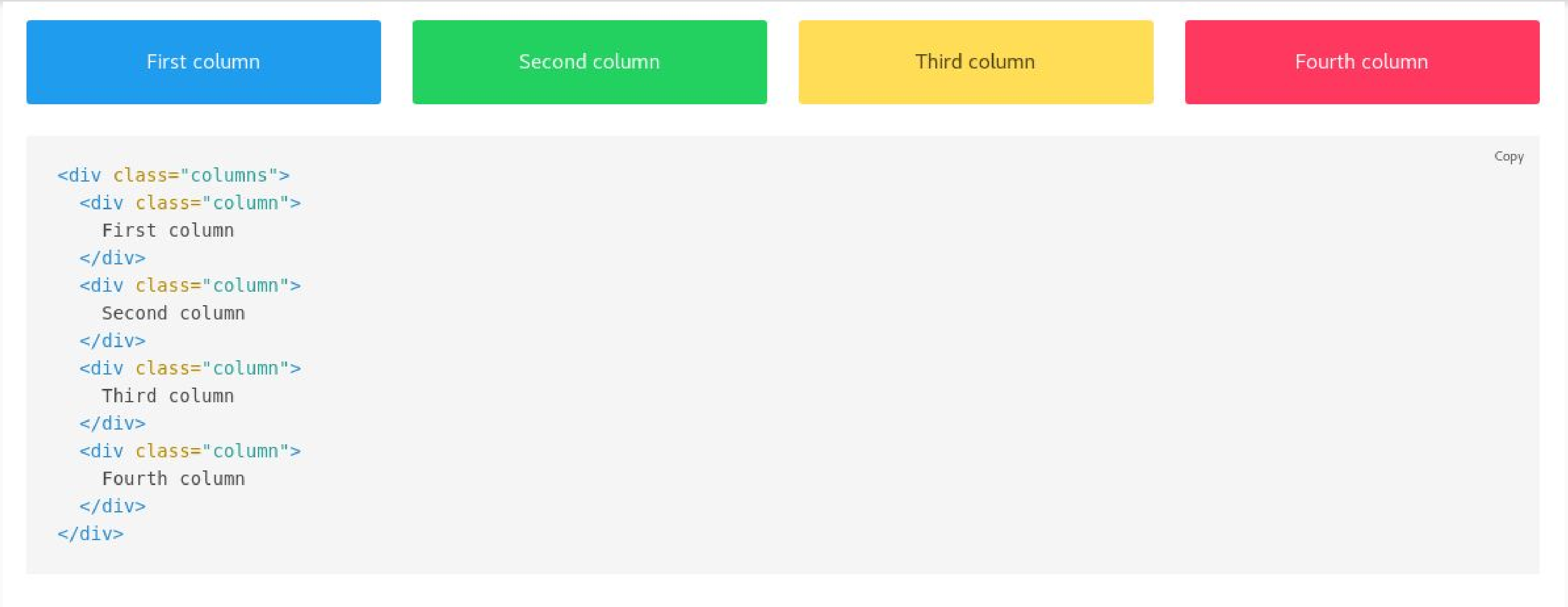
Columns

Columns

A simple way to build responsive columns

- Building a columns layout with Bulma is very simple:
  - Add a columns container
  - Add as many column elements as you want
- Each column will have an equal width, no matter the number of columns.

Basic Columns



```
<div class="columns">
  <div class="column">
    First column
  </div>
  <div class="column">
    Second column
  </div>
  <div class="column">
    Third column
  </div>
  <div class="column">
    Fourth column
  </div>
</div>
```

Column sizes

Define the size of each column individually

If you want to change the size of a single column, you can use one of the following classes:

- Is-three-quarters
- Is-two-thirds
- Is-half
- Is-one-third
- is-one-quarter

Columns responsiveness

Handle different column layouts for each breakpoint

- By default, columns are only activated from tablet onwards.
- This means columns are stacked on top of each other on mobile.
- If you want columns to work on mobile too, just add the is-mobile modifier on the columns container:

```
<div class="columns is-mobile">
	<div class="column">1</div>
	<div class="column">2</div>
	<div class="column">3</div>
	<div class="column">4</div>
</div>
```

Nesting columns

A simple way to build responsive columns

You can nest columns to have more flexibility in your design. You only need to follow this structure:

- columns: top-level columns container
    - column
        - columns: nested columns
            - column and so on…