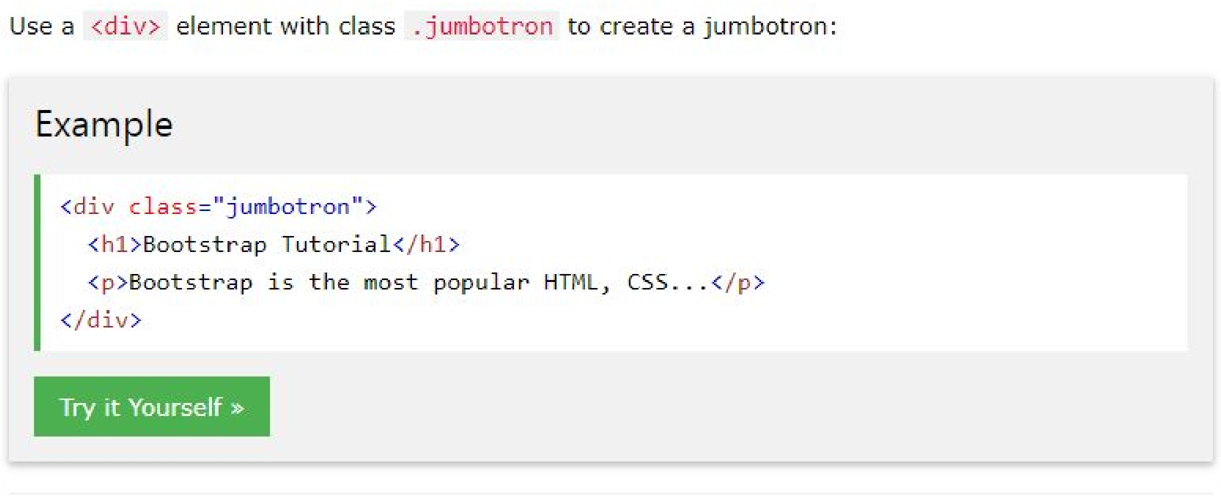
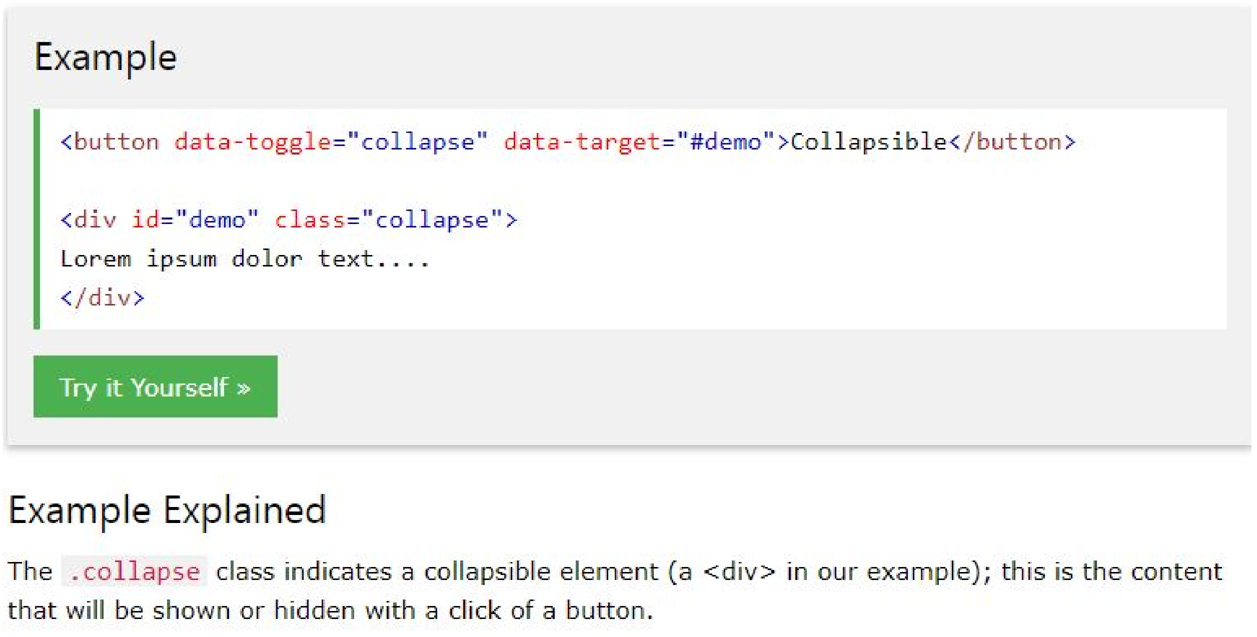
```
<div class="jumbotron jumbotron-fluid">
  <div class="container">
    <h1>Bootstrap Tutorial</h1>
    <p>Bootstrap is the most popular HTML, CSS...</p>
  </div>
</div>
```



Use a `<div>` element with class `.jumbotron` to create a jumbotron:

Example

```
<div class="jumbotron">
  <h1>Bootstrap Tutorial</h1>
  <p>Bootstrap is the most popular HTML, CSS...</p>
</div>
```



Example

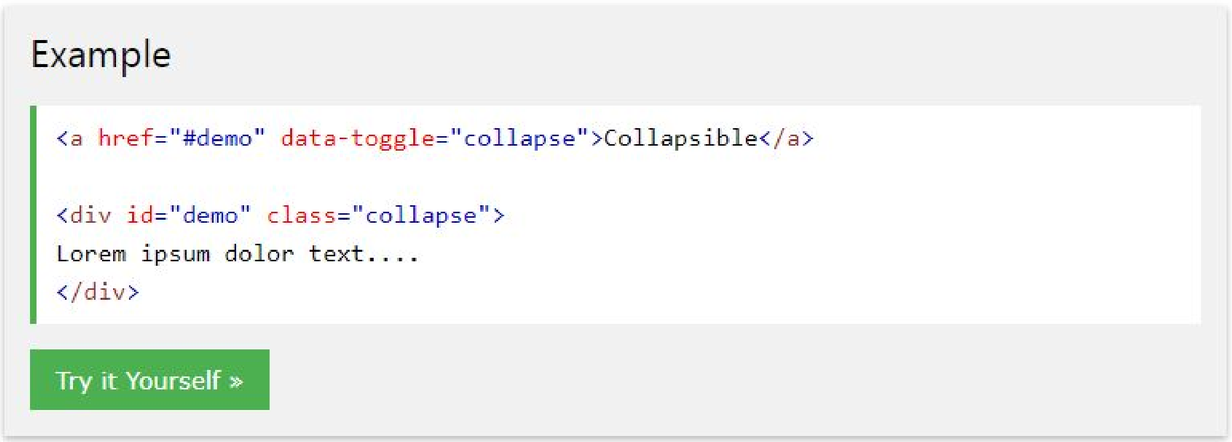
```
<button data-toggle="collapse" data-target="#demo">Collapsible</button>

<div id="demo" class="collapse">
Lorem ipsum dolor text....
</div>
```

Example Explained

The `.collapse` class indicates a collapsible element (a <div> in our example); this is the content that will be shown or hidden with a click of a button.

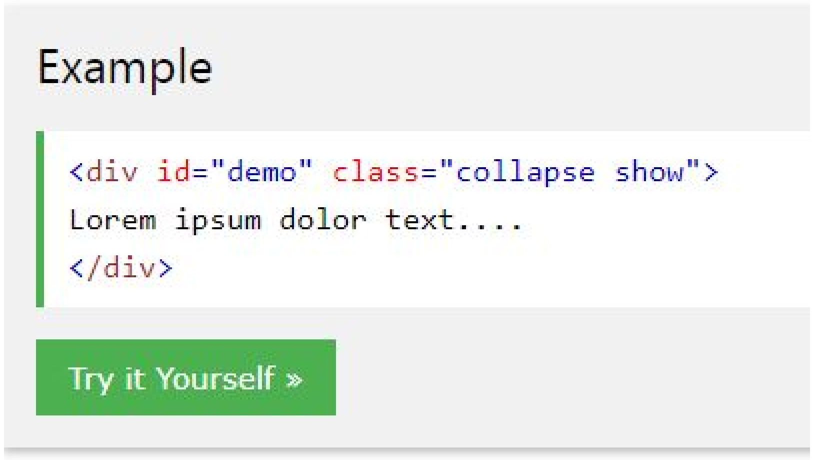
To control (show/hide) the collapsible content, add the `data-toggle="collapse"` attribute to an <a> or a <button> element. Then add the `data-target="#id"` attribute to connect the button with the collapsible content (<div id="demo">).

Note: For <a> elements, you can use the `href` attribute instead of the `data-target` attribute:


Example

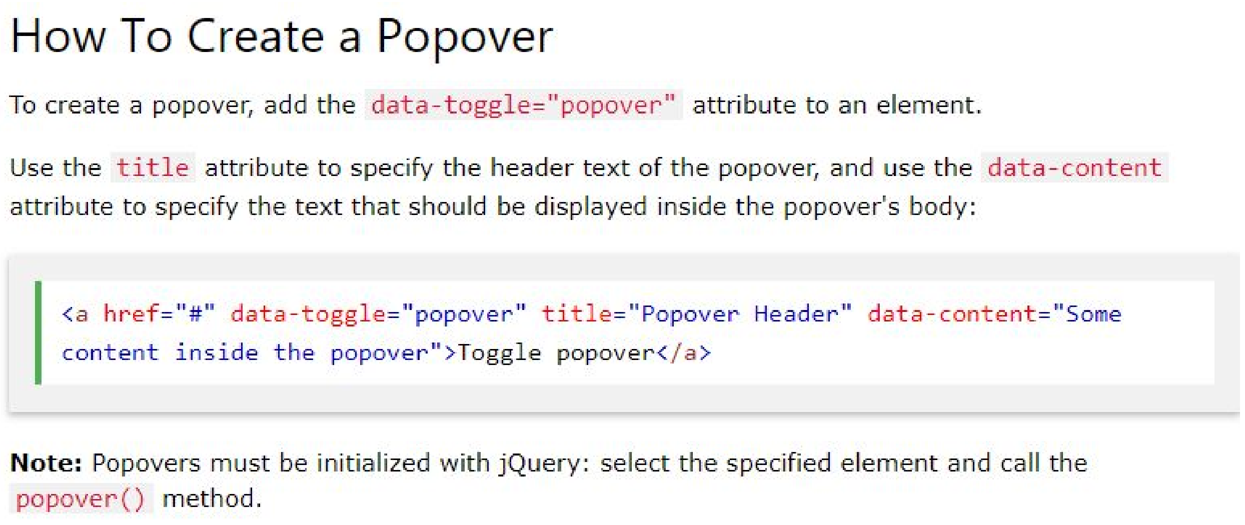
```
<a href="#demo" data-toggle="collapse">Collapsible</a>

<div id="demo" class="collapse">
Lorem ipsum dolor text....
</div>
```

By default, the collapsible content is hidden. However, you can add the `.show` class to show the content by default:


Example

```
<div id="demo" class="collapse show">
Lorem ipsum dolor text....
</div>
```

```
<div id="accordion">

  <div class="card">
    <div class="card-header">
      <a class="card-link" data-toggle="collapse"
href="#collapseOne">
     Collapsible Group Item #1
      </a>
    </div>
    <div id="collapseOne" class="collapse show"
data-parent="#accordion">
      <div class="card-body">
     Lorem ipsum..
      </div>
    </div>
  </div>

  <div class="card">
    <div class="card-header">
      <a class="collapsed card-link" data-toggle="collapse"
href="#collapseTwo">
     Collapsible Group Item #2
      </a>
    </div>
    <div id="collapseTwo" class="collapse"
data-parent="#accordion">
      <div class="card-body">
     Lorem ipsum..
      </div>
    </div>
  </div>

  <div class="card">
    <div class="card-header">
      <a class="collapsed card-link"
data-toggle="collapse"href="#collapseThree">
     Collapsible Group Item #3
      </a>
    </div>
    <div id="collapseThree" class="collapse"
data-parent="#accordion">
      <div class="card-body">
     Lorem ipsum..
      </div>
    </div>
  </div>

</div>
```



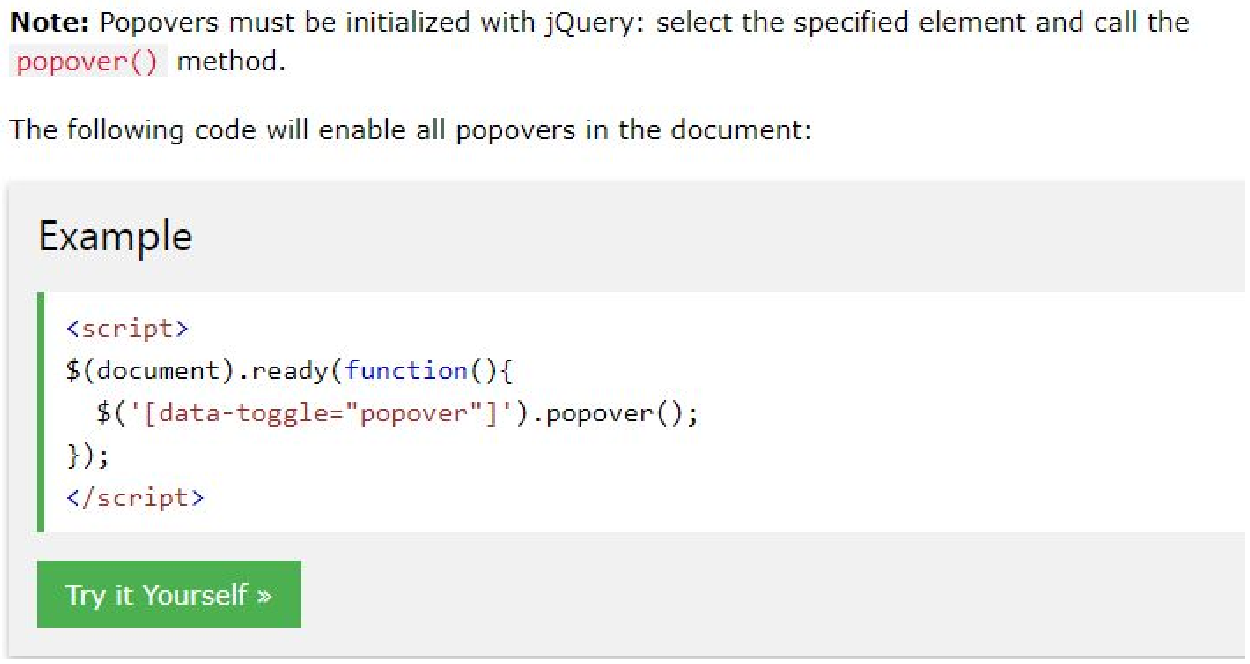
How To Create a Popover

To create a popover, add the `data-toggle="popover"` attribute to an element.

Use the `title` attribute to specify the header text of the popover, and use the `data-content` attribute to specify the text that should be displayed inside the popover's body:

```
<a href="#" data-toggle="popover" title="Popover Header" data-content="Some
content inside the popover">Toggle popover</a>
```

Note: Popovers must be initialized with jQuery: select the specified element and call the `popover()` method.



Note: Popovers must be initialized with jQuery: select the specified element and call the `popover()` method.

The following code will enable all popovers in the document:

Example

```
<script>
$(document).ready(function(){
  $('[data-toggle="popover"]').popover();
});
</script>
```

Positioning Popovers

By default, the popover will appear on the right side of the element.

Use the `data-placement` attribute to set the position of the popover on top, bottom, left or the right side of the element:

Example

```
<a href="#" title="Header" data-toggle="popover" data-placement="top" data-content="Content">Click</a>
<a href="#" title="Header" data-toggle="popover" data-placement="bottom" data-content="Content">Click</a>
<a href="#" title="Header" data-toggle="popover" data-placement="left" data-content="Content">Click</a>
<a href="#" title="Header" data-toggle="popover" data-placement="right" data-content="Content">Click</a>
```

Try it Yourself »

Closing Popovers

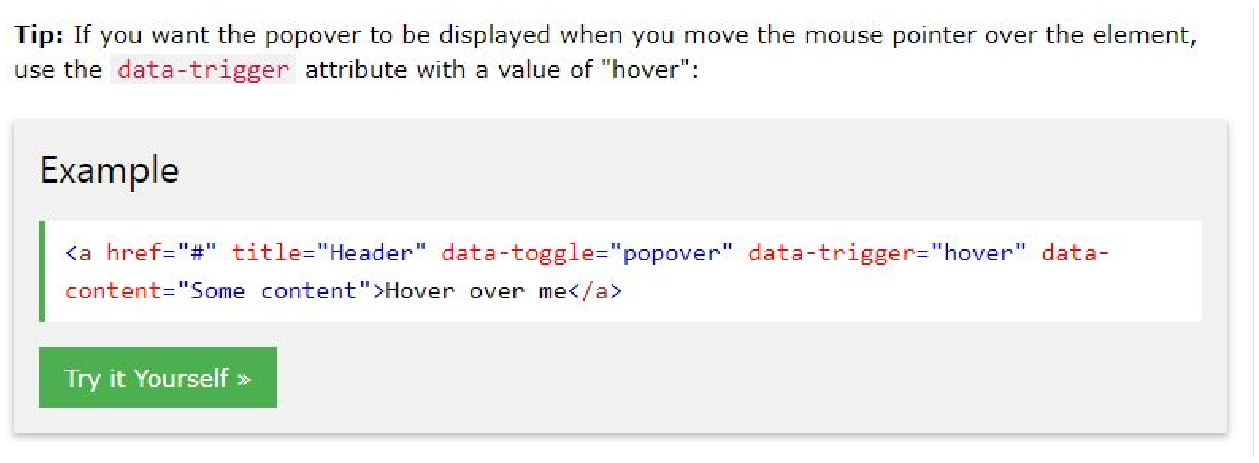
By default, the popover is closed when you click on the element again. However, you can use the `data-trigger="focus"` attribute which will close the popover when clicking outside the element:

Example

```
<a href="#" title="Dismissible popover" data-toggle="popover" data-trigger="focus" data-content="Click anywhere in the document to close this popover">Click me</a>
```

Try it Yourself »

Tip: If you want the popover to be displayed when you move the mouse pointer over the element, use the `data-trigger` attribute with a value of "hover":


Tip: If you want the popover to be displayed when you move the mouse pointer over the element, use the `data-trigger` attribute with a value of "hover":

Example

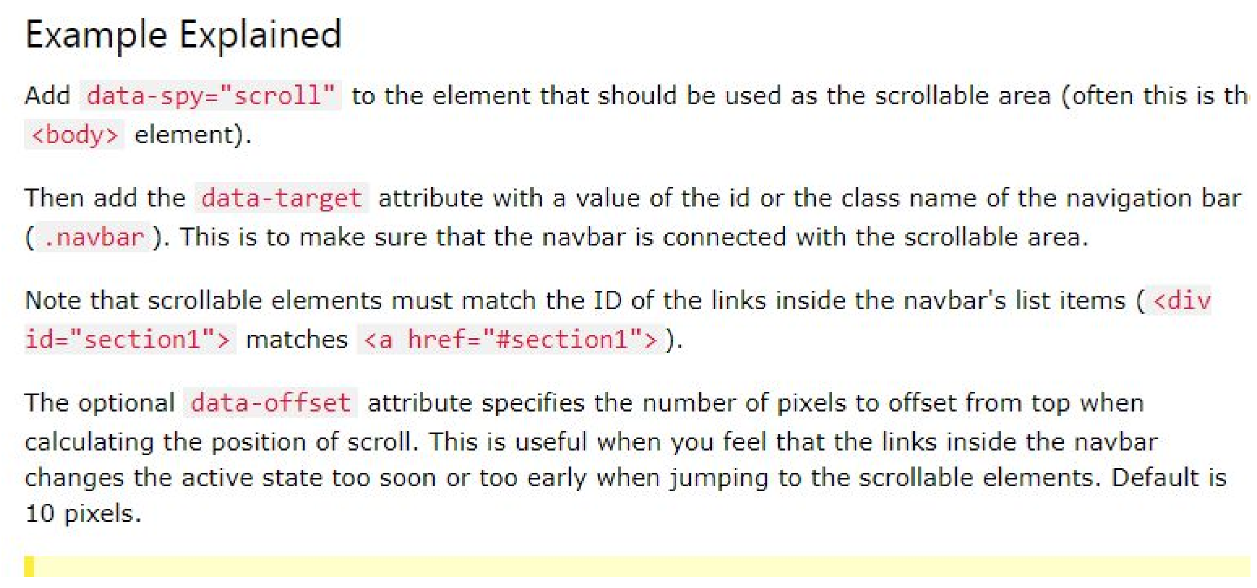
```
<a href="#" title="Header" data-toggle="popover" data-trigger="hover" data-
content="Some content">Hover over me</a>
```

SCROLL SPY

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.2.1/css/bootst
rap.min.css">
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.m
in.js"></script>
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.6/umd
/popper.min.js"></script>
  <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.2.1/js/bootstra
p.min.js"></script>
  <style>
  body {
      position: relative;
  }
  </style>
</head>
<body data-spy="scroll" data-target=".navbar" data-offset="50">
```

```
<nav class="navbar navbar-expand-sm bg-dark navbar-dark
fixed-top">
  <ul class="navbar-nav">
    <li class="nav-item">
      <a class="nav-link" href="#section1">Section 1</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#section2">Section 2</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#section3">Section 3</a>
    </li>
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#"
id="navbardrop" data-toggle="dropdown">
        Section 4
      </a>
      <div class="dropdown-menu">
        <a class="dropdown-item" href="#section41">Link 1</a>
        <a class="dropdown-item" href="#section42">Link 2</a>
      </div>
    </li>
  </ul>
</nav>

<div id="section1" class="container-fluid bg-success"
style="padding-top:70px;padding-bottom:70px">
  <h1>Section 1</h1>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
</div>
<div id="section2" class="container-fluid bg-warning"
style="padding-top:70px;padding-bottom:70px">
  <h1>Section 2</h1>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
</div>
<div id="section3" class="container-fluid bg-secondary"
style="padding-top:70px;padding-bottom:70px">
  <h1>Section 3</h1>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
</div>
<div id="section41" class="container-fluid bg-danger"
style="padding-top:70px;padding-bottom:70px">
  <h1>Section 4 Submenu 1</h1>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
</div>
<div id="section42" class="container-fluid bg-info"
style="padding-top:70px;padding-bottom:70px">
  <h1>Section 4 Submenu 2</h1>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
  <p>Try to scroll this section and look at the navigation bar
while scrolling! Try to scroll this section and look at the
navigation bar while scrolling!</p>
</div>

</body>
```

```
</html>
```



Example Explained

Add `data-spy="scroll"` to the element that should be used as the scrollable area (often this is th `<body>` element).

Then add the `data-target` attribute with a value of the id or the class name of the navigation bar (`.navbar`). This is to make sure that the navbar is connected with the scrollable area.

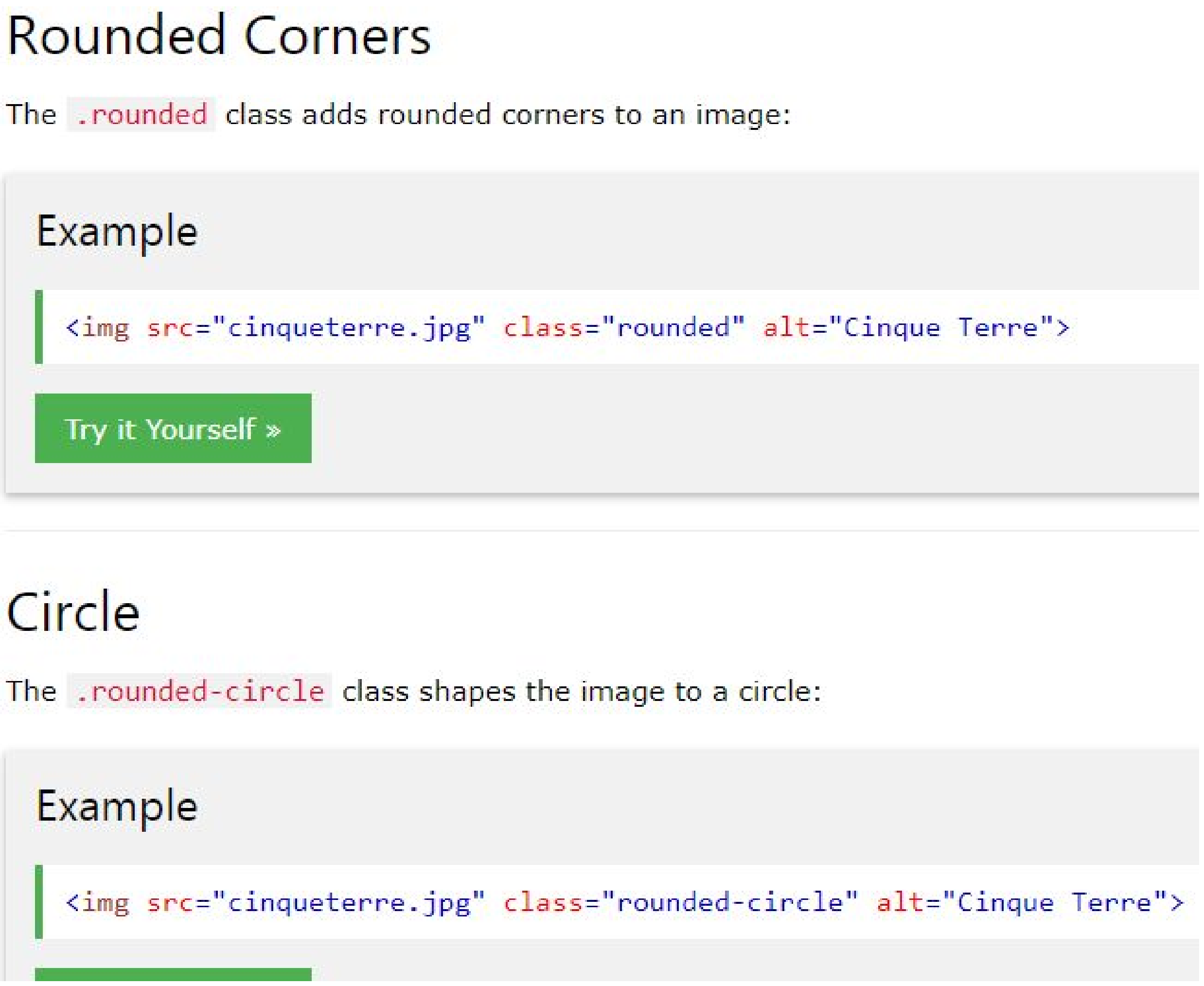
Note that scrollable elements must match the ID of the links inside the navbar's list items (`<div id="section1">` matches `<a href="#section1">`).

The optional `data-offset` attribute specifies the number of pixels to offset from top when calculating the position of scroll. This is useful when you feel that the links inside the navbar changes the active state too soon or too early when jumping to the scrollable elements. Default is 10 pixels.

TABS

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width,
initial-scale=1">
<style>
body {font-family: Arial;}

/* Style the tab */
.tab {
  overflow: hidden;
  border: 1px solid #ccc;
  background-color: #f1f1f1;
}

/* Style the buttons inside the tab */
.tab button {
  background-color: inherit;
  float: left;
  border: none;
  outline: none;
  cursor: pointer;
  padding: 14px 16px;
  transition: 0.3s;
  font-size: 17px;
}

/* Change background color of buttons on hover */
.tab button:hover {
  background-color: #ddd;
}

/* Create an active/current tablink class */
.tab button.active {
  background-color: #ccc;
}

/* Style the tab content */
.tabcontent {
  display: none;
  padding: 6px 12px;
  border: 1px solid #ccc;
  border-top: none;
}
</style>
</head>
<body>

<h2>Tabs</h2>
<p>Click on the buttons inside the tabbed menu:</p>

<div class="tab">
  <button class="tablinks" onclick="openCity(event,
'London')">London</button>
  <button class="tablinks" onclick="openCity(event,
'Paris')">Paris</button>
  <button class="tablinks" onclick="openCity(event,
'Tokyo')">Tokyo</button>
</div>

<div id="London" class="tabcontent">
```

```
  <h3>London</h3>
  <p>London is the capital city of England.</p>
</div>

<div id="Paris" class="tabcontent">
  <h3>Paris</h3>
  <p>Paris is the capital of France.</p>
</div>

<div id="Tokyo" class="tabcontent">
  <h3>Tokyo</h3>
  <p>Tokyo is the capital of Japan.</p>
</div>

<script>
function openCity(evt, cityName) {
  var i, tabcontent, tablinks;
  tabcontent = document.getElementsByClassName("tabcontent");
  for (i = 0; i < tabcontent.length; i++) {
    tabcontent[i].style.display = "none";
  }
  tablinks = document.getElementsByClassName("tablinks");
  for (i = 0; i < tablinks.length; i++) {
    tablinks[i].className = tablinks[i].className.replace("
active", "");
  }
  document.getElementById(cityName).style.display = "block";
  evt.currentTarget.className += " active";
}
</script>

</body>
</html>
```



Rounded Corners

The `.rounded` class adds rounded corners to an image:

Example

```
<img src="cinqueterre.jpg" class="rounded" alt="Cinque Terre">
```

Try it Yourself »

Circle

The `.rounded-circle` class shapes the image to a circle:

Example

```
<img src="cinqueterre.jpg" class="rounded-circle" alt="Cinque Terre">
```

The `.img-thumbnail` class shapes the image to a thumbnail (bordered):

Example

```
<img src="cinqueterre.jpg" class="img-thumbnail" alt="Cinque Terre">
```

Try it Yourself »

Aligning Images

Float an image to the right with the `.float-right` class or to the left with `.float-left`:

Centered Image

Center an image by adding the utility classes `.mx-auto` (margin:auto) and `.d-block` (display:block) to the image:

Example

```
<img src="paris.jpg" class="mx-auto d-block">
```

Responsive Images

Images come in all sizes. So do screens. Responsive images automatically adjust to fit the size of the screen.

Create responsive images by adding an `.img-fluid` class to the `<img>` tag. The image will then scale nicely to the parent element.

The `.img-fluid` class applies `max-width: 100%;` and `height: auto;` to the image:

Example

```
<img class="img-fluid" src="img_chania.jpg" alt="Chania">
```

Try it Yourself »