

[thinkadoo](#)
[Home](#)
[Get started](#)
[Scaffolding](#)
[Base CSS](#)
[Components](#)
[JavaScript](#)
[Customize](#)

[Overview](#)
[Typography](#)
[Modal](#)
[Dropdown](#)
[Scrollspy](#)
[Tab](#)
[Tooltip](#)
[Popover](#)
[Alert](#)
[Button](#)
[Collapse](#)
[Carousel](#)
[Typeahead](#)
[Affix](#)

JavaScript in Bootstrap

Individual or compiled

Plugins can be included individually (though some have required dependencies), or all at once. Both **bootstrap.js** and **bootstrap.min.js** contain all plugins in a single file.

Data attributes

You can use all Bootstrap plugins purely through the markup API without writing a single line of JavaScript. This is Bootstrap's first class API and should be your first consideration when using a plugin.

That said, in some situations it may be desirable to turn this functionality off. Therefore, we also provide the ability to disable the data attribute API by unbinding all events on the body namespaced with `data-api`. This looks like this:

```
$( 'body' ).off( 'data-api' )
```

Alternatively, to target a specific plugin, just include the plugin's name as a namespace along with the data-api namespace like this:

```
$( 'body' ).off( 'alert.data-api' )
```

Programmatic API

We also believe you should be able to use all Bootstrap plugins purely through the JavaScript API. All public APIs are single, chainable methods, and return the collection acted upon.

```
$( ".btn.danger" ).button( "toggle" ).addClass( "fat" )
```

All methods should accept an optional options object, a string which targets a particular method, or nothing (which initiates a plugin with default behavior):

```
$( "#myModal" ).modal( )           // initialized with defaults
$( "#myModal" ).modal( { keyboard: false } ) // initialized with no keyboard
$( "#myModal" ).modal( 'show' )    // initializes and invokes show immediately
```

Each plugin also exposes its raw constructor on a `Constructor` property: `$.fn.popover.Constructor`. If you'd like to get a particular plugin instance, retrieve it directly from an element: `$('[rel=popover]').data('popover')`.

Events

Bootstrap provides custom events for most plugin's unique actions. Generally, these come in an infinitive and past participle form - where the infinitive (ex. `show`) is triggered at the start of an event, and its past participle form (ex. `shown`) is trigger on the completion of an action.

All infinitive events provide `preventDefault` functionality. This provides the ability to stop the execution of an action before it starts.

```
$( '#myModal' ).on( 'show', function (e) {
    if (!data) return e.preventDefault() // stops modal from being shown
})
```

Transitions bootstrap-transition.js

About transitions

For simple transition effects, include bootstrap-transition.js once alongside the other JS files. If you're using the compiled (or minified) bootstrap.js, there is no need to include this—it's already there.

Use cases

A few examples of the transition plugin:

- Sliding or fading in modals
- Fading out tabs
- Fading out alerts
- Sliding carousel panes

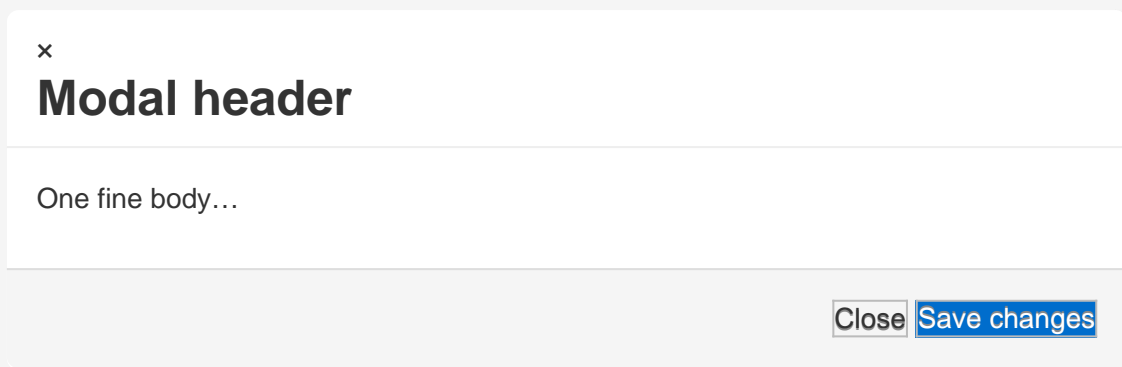
Modals bootstrap-modal.js

Examples

Modals are streamlined, but flexible, dialog prompts with the minimum required functionality and smart defaults.

Static example

A rendered modal with header, body, and set of actions in the footer.



```
<div class="modal hide fade">
  <div class="modal-header">
    <button type="button" class="close" data-dismiss="modal "
aria-hidden="true">&times;</button>
    <h3>Modal header</h3>
  </div>
  <div class="modal-body">
    <p>One fine body...</p>
  </div>
  <div class="modal-footer">
    <a href="#" class="btn">Close</a>
    <a href="#" class="btn btn-primary">Save changes</a>
  </div>
</div>
```

Live demo

Toggle a modal via JavaScript by clicking the button below. It will slide down and fade in from the top of the page.

Launch demo modal

```
<!-- Button to trigger modal -->
<a href="#myModal" role="button" class="btn" data-toggle="modal">Launch demo modal</a>

<!-- Modal -->
<div id="myModal" class="modal hide fade" tabindex="-1" role="dialog"
aria-labelledby="myModalLabel" aria-hidden="true">
  <div class="modal-header">
    <button type="button" class="close" data-dismiss="modal"
aria-hidden="true">×</button>
    <h3 id="myModalLabel">Modal header</h3>
  </div>
  <div class="modal-body">
    <p>One fine body...</p>
  </div>
  <div class="modal-footer">
    <button class="btn" data-dismiss="modal" aria-hidden="true">Close</button>
    <button class="btn btn-primary">Save changes</button>
  </div>
</div>
```

Usage

Via data attributes

Activate a modal without writing JavaScript. Set `data-toggle="modal"` on a controller element, like a button, along with a `data-target="#foo"` or `href="#foo"` to target a specific modal to toggle.

```
<button type="button" data-toggle="modal" data-target="#myModal">Launch modal</button>
```

Via JavaScript

Call a modal with id `myModal` with a single line of JavaScript:

```
$( '#myModal' ).modal(options)
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-backdrop=""`.

Name	type	default	description
backdrop	boolean	true	Includes a modal-backdrop element. Alternatively, specify <code>static</code> for a backdrop which doesn't close the modal on click.
keyboard	boolean	true	Closes the modal when escape key is pressed

Name	type	default	description
show	boolean	true	Shows the modal when initialized.
remote	path	false	If a remote url is provided, content will be loaded via jQuery's <code>load</code> method and injected into the <code>.modal-body</code> . If you're using the data api, you may alternatively use the <code>href</code> tag to specify the remote source. An example of this is shown below: <pre><a data-toggle="modal" href="remote.html" data-target="#modal">click me</pre>

Methods

`.modal(options)`

Activates your content as a modal. Accepts an optional options `object`.

```
$( '#myModal' ).modal({
  keyboard: false
})
```

`.modal('toggle')`

Manually toggles a modal.

```
$( '#myModal' ).modal( 'toggle' )
```

`.modal('show')`

Manually opens a modal.

```
$( '#myModal' ).modal( 'show' )
```

`.modal('hide')`

Manually hides a modal.

```
$( '#myModal' ).modal( 'hide' )
```

Events

Bootstrap's modal class exposes a few events for hooking into modal functionality.

Event	Description
show	This event fires immediately when the <code>show</code> instance method is called.
shown	This event is fired when the modal has been made visible to the user (will wait for css transitions to complete).
hide	This event is fired immediately when the <code>hide</code> instance method has been called.
hidden	This event is fired when the modal has finished being hidden from the user (will wait for css transitions to complete).

```
$('#myModal').on('hidden', function () {  
  // do something...  
})
```

Dropdowns bootstrap-dropdown.js

Examples

Add dropdown menus to nearly anything with this simple plugin, including the navbar, tabs, and pills.

Within a navbar

[Project Name](#)

[Dropdown](#)[Dropdown 2](#)[Dropdown 3](#)

Within tabs

- [Regular link](#)
- [Dropdown](#)
- [Dropdown 2](#)
- [Dropdown 3](#)

Usage

Via data attributes

Add `data-toggle="dropdown"` to a link or button to toggle a dropdown.

```
<div class="dropdown">
  <a class="dropdown-toggle" data-toggle="dropdown" href="#">Dropdown trigger</a>
  <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
    ...
  </ul>
</div>
```

To keep URLs intact, use the `data-target` attribute instead of `href="#"`.

```
<div class="dropdown">
  <a class="dropdown-toggle" id="dLabel" role="button" data-toggle="dropdown"
data-target="#" href="/page.html">
    Dropdown
    <b class="caret"></b>
  </a>
  <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
    ...
  </ul>
</div>
```

Via JavaScript

Call the dropdowns via JavaScript:

```
$( '.dropdown-toggle' ).dropdown()
```

Options

None

Methods

`$.dropdown('toggle')`

A programatic api for toggling menus for a given navbar or tabbed navigation.

ScrollSpy bootstrap-scrollspy.js

Example in navbar

The ScrollSpy plugin is for automatically updating nav targets based on scroll position. Scroll the area below the navbar and watch the active class change. The dropdown sub items will be highlighted as well.

Project Name
@fat

@fat

Ad leggings keytar, brunch id art party dolor labore. Pitchfork yr enim lo-fi before they sold out qui. Tumblr farm-to-table bicycle rights whatever. Anim keffiyeh carles cardigan. Velit seitan mcsweeney's photo booth 3 wolf moon irure. Cosby sweater lomo jean shorts, williamsburg hoodie minim qui you probably haven't heard of them et cardigan trust fund culpa biodiesel wes anderson aesthetic. Nihil tattooed accusamus, cred irony biodiesel keffiyeh artisan ullamco consequat.

@mdo

Veniam marfa mustache skateboard, adipisicing fugiat velit pitchfork beard. Freegan beard aliqua cupidatat mcsweeney's vero. Cupidatat four loko nisi, ea helvetica nulla carles. Tattooed cosby sweater food truck, mcsweeney's quis non freegan vinyl. Lo-fi wes anderson +1 sartorial. Carles non aesthetic exercitation quis gentrify. Brooklyn adipisicing craft beer vice keytar deserunt.

one

Occaecat commodo aliqua delectus. Fap craft beer deserunt skateboard ea. Lomo bicycle rights adipisicing banh mi, velit ea sunt next level locavore single-origin coffee in magna veniam. High life id vinyl, echo park consequat quis aliquip banh mi pitchfork. Vero VHS est adipisicing. Consectetur nisi DIY minim messenger bag. Cred ex in, sustainable delectus consectetur fanny pack iphone.

two

In incididunt echo park, officia deserunt mcsweeney's proident master cleanse thundercats sapiente veniam. Excepteur VHS elit, proident shoreditch +1 biodiesel laborum craft beer. Single-origin coffee wayfarers irure four loko, cupidatat terry richardson master cleanse. Assumenda you probably haven't heard of them art party fanny pack, tattooed nulla cardigan tempor ad. Proident wolf nesciunt sartorial keffiyeh eu banh mi sustainable. Elit wolf voluptate, lo-fi ea portland before they sold out four loko. Locavore enim nostrud mlkshk brooklyn nesciunt.

three

Ad leggings keytar, brunch id art party dolor labore. Pitchfork yr enim lo-fi before they sold out qui. Tumblr farm-to-table bicycle rights whatever. Anim keffiyeh carles cardigan. Velit seitan mcsweeney's photo booth 3 wolf moon irure. Cosby sweater lomo jean shorts, williamsburg hoodie minim qui you probably haven't heard of them et cardigan trust fund culpa biodiesel wes anderson aesthetic. Nihil tattooed accusamus, cred irony biodiesel keffiyeh artisan ullamco consequat.

Keytar twee blog, culpa messenger bag marfa whatever delectus food truck. Sapiente synth id assumenda. Locavore sed helvetica cliché irony, thundercats you probably haven't heard of them consequat hoodie gluten-free lo-fi fap aliquip. Labore elit placeat before they sold out, terry richardson proident brunch nesciunt quis cosby sweater pariatur keffiyeh ut helvetica artisan. Cardigan craft beer seitan ready-made velit. VHS chambray laboris tempor veniam. Anim mollit minim commodo ullamco thundercats.

Via data attributes

To easily add scrollspy behavior to your topbar navigation, just add `data-spy="scroll"` to the element you want to spy on (most typically this would be the body) and `data-target=".navbar"` to select which nav to use. You'll want to use scrollspy with a `.nav` component.

```
<body data-spy="scroll" data-target=".navbar">...</body>
```

Via JavaScript

Call the scrollspy via JavaScript:

```
$( '#navbar' ).scrollspy()
```

Heads up! Navbar links must have resolvable id targets. For example, a `home` must correspond to something in the dom like `<div id="home"></div>`.

Methods

.scrollspy('refresh')

When using scrollspy in conjunction with adding or removing of elements from the DOM, you'll need to call the refresh method like so:

```
$( '[data-spy="scroll"]' ).each(function () {  
    var $spy = $(this).scrollspy('refresh')  
});
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-offset=""`.

Name	type	default	description
offset	number	10	Pixels to offset from top when calculating position of scroll.

Events

Event	Description
activate	This event fires whenever a new item becomes activated by the scrollspy.

Togglable tabs bootstrap-tab.js

Example tabs

Add quick, dynamic tab functionality to transition through panes of local content, even via dropdown menus.

Home
Profile
Dropdown

Raw denim you probably haven't heard of them jean shorts Austin. Nesciunt tofu stumptown aliqua, retro synth master cleanse. Mustache cliche tempor, williamsburg carles vegan helvetica. Reprehenderit butcher retro keffiyeh dreamcatcher synth. Cosby sweater eu banh mi, qui irure terry richardson ex squid. Aliquip placeat salvia cillum iphone. Seitan aliquip quis cardigan american apparel, butcher voluptate nisi qui.

Food truck fixie locavore, accusamus mcsweeney's marfa nulla single-origin coffee squid. Exercitation +1 labore velit, blog sartorial PBR leggings next level wes anderson artisan four loko farm-to-table craft beer twee. Qui photo booth letterpress, commodo enim craft beer mlkshk aliquip jean shorts ullamco ad vinyl cillum PBR. Homo nostrud organic, assumenda labore aesthetic magna delectus mollit. Keytar helvetica VHS salvia yr, vero magna velit sapiente labore stumptown. Vegan fanny pack odio cillum wes anderson 8-bit, sustainable jean shorts beard ut DIY ethical culpa terry richardson biodiesel. Art party scenester stumptown, tumblr butcher vero sint qui sapiente accusamus tattooed echo park.

Etsy mixtape wayfarers, ethical wes anderson tofu before they sold out mcsweeney's organic lomo retro fanny pack lo-fi farm-to-table readymade. Messenger bag gentrify pitchfork tattooed craft beer, iphone skateboard locavore carles etsy salvia banksy hoodie helvetica. DIY synth PBR banksy irony. Leggings gentrify squid 8-bit cred pitchfork. Williamsburg banh mi whatever gluten-free, carles pitchfork biodiesel fixie etsy retro mlkshk vice blog. Scenester cred you probably haven't heard of them, vinyl craft beer blog stumptown. Pitchfork sustainable tofu synth chambray yr.

Trust fund seitan letterpress, keytar raw denim keffiyeh etsy art party before they sold out master cleanse gluten-free squid scenester freegan cosby sweater. Fanny pack portland seitan DIY, art party locavore wolf cliche high life echo park Austin. Cred vinyl keffiyeh DIY salvia PBR, banh mi before they sold out farm-to-table VHS viral locavore cosby sweater. Lomo wolf viral, mustache readymade thundercats keffiyeh craft beer marfa ethical. Wolf salvia freegan, sartorial keffiyeh echo park vegan.

Usage

Enable tabbable tabs via JavaScript (each tab needs to be activated individually):

```
$( '#myTab a' ).click(function (e) {  
  e.preventDefault();  
  $(this).tab('show');  
})
```

You can activate individual tabs in several ways:

```
$('#myTab a[href="#profile"]').tab('show'); // Select tab by name
$('#myTab a:first').tab('show'); // Select first tab
$('#myTab a:last').tab('show'); // Select last tab
$('#myTab li:eq(2) a').tab('show'); // Select third tab (0-indexed)
```

Markup

You can activate a tab or pill navigation without writing any JavaScript by simply specifying `data-toggle="tab"` or `data-toggle="pill"` on an element. Adding the `nav` and `nav-tabs` classes to the tab `ul` will apply the Bootstrap tab styling.

```
<ul class="nav nav-tabs">
  <li><a href="#home" data-toggle="tab">Home</a></li>
  <li><a href="#profile" data-toggle="tab">Profile</a></li>
  <li><a href="#messages" data-toggle="tab">Messages</a></li>
  <li><a href="#settings" data-toggle="tab">Settings</a></li>
</ul>
```

Methods

\$.tab

Activates a tab element and content container. Tab should have either a `data-target` or an `href` targeting a container node in the DOM.

```
<ul class="nav nav-tabs" id="myTab">
  <li class="active"><a href="#home">Home</a></li>
  <li><a href="#profile">Profile</a></li>
  <li><a href="#messages">Messages</a></li>
  <li><a href="#settings">Settings</a></li>
</ul>
```

```
<div class="tab-content">
  <div class="tab-pane active" id="home">...</div>
  <div class="tab-pane" id="profile">...</div>
  <div class="tab-pane" id="messages">...</div>
  <div class="tab-pane" id="settings">...</div>
</div>
```

```
<script>
  $(function () {
    $('#myTab a:last').tab('show');
  })
</script>
```

Events

Event	Description
show	This event fires on tab show, but before the new tab has been shown. Use <code>event.target</code> and <code>event.relatedTarget</code> to target the active tab and the previous active tab (if available) respectively.
shown	This event fires on tab show after a tab has been shown. Use <code>event.target</code> and <code>event.relatedTarget</code> to target the active tab and the previous active tab (if available) respectively.

```
$( 'a[data-toggle="tab"]' ).on( 'shown', function (e) {
  e.target // activated tab
  e.relatedTarget // previous tab
})
```

Tooltips bootstrap-tooltip.js

Examples

Inspired by the excellent jQuery.tipsy plugin written by Jason Frame; Tooltips are an updated version, which don't rely on images, use CSS3 for animations, and data-attributes for local title storage.

Hover over the links below to see tooltips:

Tight pants next level keffiyeh [you probably](#) haven't heard of them. Photo booth beard raw denim letterpress vegan messenger bag stumptown. Farm-to-table seitan, mcsweeney's fixie sustainable quinoa 8-bit american apparel [have a](#) terry richardson vinyl chambray. Beard stumptown, cardigans banh mi lomo thundercats. Tofu biodiesel williamsburg marfa, four loko mcsweeney's cleanse vegan chambray. A really ironic artisan [whatever keytar](#), scenester farm-to-table banksy Austin [twitter handle](#) freegan cred raw denim single-origin coffee viral.

Four directions

[Tooltip on top](#)
[Tooltip on right](#)
[Tooltip on bottom](#)
[Tooltip on left](#)

Usage

Trigger the tooltip via JavaScript:

```
$('#example').tooltip(options)
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-animation=""`.

Name	type	default	description
animation	boolean	true	apply a css fade transition to the tooltip
html	boolean	false	Insert html into the tooltip. If false, jquery's <code>text</code> method will be used to insert content into the dom. Use text if you're worried about XSS attacks.
placement	string function	'top'	how to position the tooltip - top bottom left right
selector	string	false	If a selector is provided, tooltip objects will be delegated to the specified targets.
title	string function	"	default title value if `title` tag isn't present
trigger	string	'hover'	how tooltip is triggered - click hover focus manual
delay	number object	0	delay showing and hiding the tooltip (ms) - does not apply to manual trigger type If a number is supplied, delay is applied to both hide/show Object structure is: <code>delay: { show: 500, hide: 100 }</code>

Heads up! Options for individual tooltips can alternatively be specified through the use of data attributes.

Markup

For performance reasons, the Tooltip and Popover data-apis are opt in. If you would like to use them just specify a selector option.

```
<a href="#" rel="tooltip" title="first tooltip">hover over me</a>
```

Methods

`$.tooltip(options)`

Attaches a tooltip handler to an element collection.

`.tooltip('show')`

Reveals an element's tooltip.

```
$('#element').tooltip('show')
```

`.tooltip('hide')`

Hides an element's tooltip.

```
$('#element').tooltip('hide')
```

`.tooltip('toggle')`

Toggles an element's tooltip.

```
$('#element').tooltip('toggle')
```

`.tooltip('destroy')`

Hides and destroys an element's tooltip.

```
$('#element').tooltip('destroy')
```

Popovers `bootstrap-popover.js`

Examples

Add small overlays of content, like those on the iPad, to any element for housing secondary information. Hover over the button to trigger the popover. **Requires [Tooltip](#) to be included.**

Static popover

Four options are available: top, right, bottom, and left aligned.

Popover top

Sed posuere consectetur est at lobortis. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum.

Popover right

Sed posuere consectetur est at lobortis. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum.

Popover bottom

Sed posuere consectetur est at lobortis. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum.

Popover left

Sed posuere consectetur est at lobortis. Aenean eu leo quam. Pellentesque ornare sem lacinia quam venenatis vestibulum.

No markup shown as popovers are generated from JavaScript and content within a `data` attribute.

Live demo

Click to toggle popover

Four directions

Popover on top
Popover on right
Popover on bottom
Popover on left

Usage

Enable popovers via JavaScript:

```
$('#example').popover(options)
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-animation=""`.

Name	type	default	description
animation	boolean	true	apply a css fade transition to the tooltip
html	boolean	false	Insert html into the popover. If false, jquery's <code>text</code> method will be used to insert content into the dom. Use text if you're worried about XSS attacks.
placement	string function	'right'	how to position the popover - top bottom left right
selector	string	false	if a selector is provided, tooltip objects will be delegated to the specified targets
trigger	string	'click'	how popover is triggered - click hover focus manual
title	string function	"	default title value if `title` attribute isn't present

Name	type	default	description
content	string function	"	default content value if `data-content` attribute isn't present
delay	number object	0	delay showing and hiding the popover (ms) - does not apply to manual trigger type If a number is supplied, delay is applied to both hide/show Object structure is: <code>delay: { show: 500, hide: 100 }</code>

Heads up! Options for individual popovers can alternatively be specified through the use of data attributes.

Markup

For performance reasons, the Tooltip and Popover data-apis are opt in. If you would like to use them just specify a selector option.

Methods

`$.popover(options)`

Initializes popovers for an element collection.

`.popover('show')`

Reveals an elements popover.

```
$('#element').popover('show')
```

`.popover('hide')`

Hides an elements popover.

```
$('#element').popover('hide')
```

`.popover('toggle')`

Toggles an elements popover.

```
$('#element').popover('toggle')
```

`.popover('destroy')`

Hides and destroys an element's popover.

```
$('#element').popover('destroy')
```

Alert messages bootstrap-alert.js

Example alerts

Add dismiss functionality to all alert messages with this plugin.

× **Holy guacamole!** Best check yo self, you're not looking too good.

×

Oh snap! You got an error!

Change this and that and try again. Duis mollis, est non commodo luctus, nisi erat porttitor ligula, eget lacinia odio sem nec elit. Cras mattis consectetur purus sit amet fermentum.

Take this action Or do this

Usage

Enable dismissal of an alert via JavaScript:

```
$(".alert").alert()
```

Markup

Just add `data-dismiss="alert"` to your close button to automatically give an alert close functionality.

```
<a class="close" data-dismiss="alert" href="#">&times;</a>
```

Methods

`$.alert()`

Wraps all alerts with close functionality. To have your alerts animate out when closed, make sure they have the `.fade` and `.in` class already applied to them.

`.alert('close')`

Closes an alert.

```
$(".alert").alert('close')
```

Events

Bootstrap's alert class exposes a few events for hooking into alert functionality.

Event	Description
close	This event fires immediately when the <code>close</code> instance method is called.
closed	This event is fired when the alert has been closed (will wait for css transitions to complete).

```
$('#my-alert').bind('closed', function () {  
    // do something..  
})
```

Buttons `bootstrap-button.js`

Example uses

Do more with buttons. Control button states or create groups of buttons for more components like toolbars.

Stateful

Add `data-loading-text="Loading..."` to use a loading state on a button.

Loading state

```
<button type="button" class="btn btn-primary" data-loading-text="Loading...">Loading  
state</button>
```

Single toggle

Add `data-toggle="button"` to activate toggling on a single button.

Single Toggle

```
<button type="button" class="btn" data-toggle="button">Single Toggle</button>
```

Checkbox

Add `data-toggle="buttons-checkbox"` for checkbox style toggling on `btn-group`.

Left Middle Right

```
<div class="btn-group" data-toggle="buttons-checkbox">
  <button type="button" class="btn">Left</button>
  <button type="button" class="btn">Middle</button>
  <button type="button" class="btn">Right</button>
</div>
```

Radio

Add data-toggle="buttons-radio" for radio style toggling on btn-group.

Left Middle Right

```
<div class="btn-group" data-toggle="buttons-radio">
  <button type="button" class="btn">Left</button>
  <button type="button" class="btn">Middle</button>
  <button type="button" class="btn">Right</button>
</div>
```

Usage

Enable buttons via JavaScript:

```
$( '.nav-tabs' ).button( )
```

Markup

Data attributes are integral to the button plugin. Check out the example code below for the various markup types.

Options

None

Methods

`$.button('toggle')`

Toggles push state. Gives the button the appearance that it has been activated.

Heads up! You can enable auto toggling of a button by using the `data-toggle` attribute.

```
<button type="button" class="btn" data-toggle="button" >...</button>
```

`$.button('loading')`

Sets button state to loading - disables button and swaps text to loading text. Loading text should be defined on the button element using the data attribute `data-loading-text`.

```
<button type="button" class="btn" data-loading-text="loading stuff..." >...</button>
```

Heads up! Firefox persists the disabled state across page loads. A workaround for this is to use `autocomplete="off"`.

`$.button('reset')`

Resets button state - swaps text to original text.

`$.button(string)`

Resets button state - swaps text to any data defined text state.

```
<button type="button" class="btn" data-complete-text="finished!" >...</button>
<script>
  $(''.btn').button('complete')
</script>
```

Collapse bootstrap-collapse.js

About

Get base styles and flexible support for collapsible components like accordions and navigation.

* Requires the Transitions plugin to be included.

Example accordion

Using the collapse plugin, we built a simple accordion style widget:

Collapsible Group Item #1

Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad squid. 3 wolf moon officia aute, non cupidatat skateboard dolor brunch. Food truck quinoa nesciunt laborum eiusmod. Brunch 3 wolf moon tempor, sunt aliqua put a bird on it squid single-origin coffee nulla assumenda shoreditch et. Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea proident. Ad vegan excepteur butcher vice lomo. Leggings occaecat craft beer farm-to-table, raw denim aesthetic synth nesciunt you probably haven't heard of them accusamus

labore sustainable VHS.

Collapsible Group Item #2

Anim pariatur cliché reprehenderit, enim eiusmod high life accusamus terry richardson ad squid. 3 wolf moon officia aute, non cupidatat skateboard dolor brunch. Food truck quinoa nesciunt laborum eiusmod. Brunch 3 wolf moon tempor, sunt aliqua put a bird on it squid single-origin coffee nulla assumenda shoreditch et. Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea proident. Ad vegan excepteur butcher vice lomo. Leggings occaecat craft beer farm-to-table, raw denim aesthetic synth nesciunt you probably haven't heard of them accusamus labore sustainable VHS.

Collapsible Group Item #3

Anim pariatur cliché reprehenderit, enim eiusmod high life accusamus terry richardson ad squid. 3 wolf moon officia aute, non cupidatat skateboard dolor brunch. Food truck quinoa nesciunt laborum eiusmod. Brunch 3 wolf moon tempor, sunt aliqua put a bird on it squid single-origin coffee nulla assumenda shoreditch et. Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea proident. Ad vegan excepteur butcher vice lomo. Leggings occaecat craft beer farm-to-table, raw denim aesthetic synth nesciunt you probably haven't heard of them accusamus labore sustainable VHS.

```

<div class="accordion" id="accordion2">
  <div class="accordion-group">
    <div class="accordion-heading">
      <a class="accordion-toggle" data-toggle="collapse" data-parent="#accordion2"
href="#collapseOne">
        Collapsible Group Item #1
      </a>
    </div>
    <div id="collapseOne" class="accordion-body collapse in">
      <div class="accordion-inner">
        Anim pariatur cliche...
      </div>
    </div>
  </div>
  <div class="accordion-group">
    <div class="accordion-heading">
      <a class="accordion-toggle" data-toggle="collapse" data-parent="#accordion2"
href="#collapseTwo">
        Collapsible Group Item #2
      </a>
    </div>
    <div id="collapseTwo" class="accordion-body collapse">
      <div class="accordion-inner">
        Anim pariatur cliche...
      </div>
    </div>
  </div>
</div>
...

```

You can also use the plugin without the accordion markup. Make a button toggle the expanding and collapsing of another element.

```

<button type="button" class="btn btn-danger" data-toggle="collapse"
data-target="#demo">
  simple collapsible
</button>

<div id="demo" class="collapse in"> ... </div>

```

Usage

Via data attributes

Just add `data-toggle="collapse"` and a `data-target` to element to automatically assign control of a collapsible element. The `data-target` attribute accepts a css selector to apply the collapse to. Be sure to add the class

`collapse` to the collapsible element. If you'd like it to default open, add the additional class `in`.

To add accordion-like group management to a collapsible control, add the data attribute `data-parent="#selector"`. Refer to the demo to see this in action.

Via JavaScript

Enable manually with:

```
$( ".collapse" ).collapse( )
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-parent=""`.

Name	type	default	description
parent	selector	false	If selector then all collapsible elements under the specified parent will be closed when this collapsible item is shown. (similar to traditional accordion behavior)
toggle	boolean	true	Toggles the collapsible element on invocation

Methods

`.collapse(options)`

Activates your content as a collapsible element. Accepts an optional options `object`.

```
$( '#myCollapsible' ).collapse({  
  toggle: false  
})
```

`.collapse('toggle')`

Toggles a collapsible element to shown or hidden.

`.collapse('show')`

Shows a collapsible element.

`.collapse('hide')`

Hides a collapsible element.

Events

Bootstrap's collapse class exposes a few events for hooking into collapse functionality.

Event	Description
show	This event fires immediately when the <code>show</code> instance method is called.
shown	This event is fired when a collapse element has been made visible to the user (will wait for css transitions to complete).

Event	Description
hide	This event is fired immediately when the <code>hide</code> method has been called.
hidden	This event is fired when a collapse element has been hidden from the user (will wait for css transitions to complete).

```
$('#myCollapsible').on('hidden', function () {
    // do something...
})
```

Carousel bootstrap-carousel.js

Example carousel

The slideshow below shows a generic plugin and component for cycling through elements like a carousel.



First Thumbnail label

Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta gravida at eget metus. Nullam id dolor id nibh ultricies vehicula ut id elit.



Second Thumbnail label

Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta gravida at eget metus. Nullam id dolor id nibh ultricies vehicula ut id elit.



Third Thumbnail label

Cras justo odio, dapibus ac facilisis in, egestas eget quam. Donec id elit non mi porta gravida at eget metus. Nullam id dolor id nibh ultricies vehicula ut id elit.



```
<div id="myCarousel" class="carousel slide">
  <!-- Carousel items -->
  <div class="carousel-inner">
    <div class="active item">...</div>
    <div class="item">...</div>
    <div class="item">...</div>
  </div>
  <!-- Carousel nav -->
  <a class="carousel-control left" href="#myCarousel" data-slide="prev">&lsaquo;</a>
  <a class="carousel-control right" href="#myCarousel" data-slide="next">&rsaquo;</a>
</div>
```

Heads up! When implementing this carousel, remove the images we have provided and replace them with your own.

Usage

Via data attributes

...

Via JavaScript

Call carousel manually with:

```
$('.carousel').carousel()
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-interval=""`.

Name	type	default	description
interval	number	5000	The amount of time to delay between automatically cycling an item. If false, carousel will not automatically cycle.
pause	string	"hover"	Pauses the cycling of the carousel on mouseenter and resumes the cycling of the carousel on mouseleave.

Methods

`.carousel(options)`

initializes the carousel with an optional options `object` and starts cycling through items.

```
$('.carousel').carousel({  
  interval: 2000  
})
```

`.carousel('cycle')`

Cycles through the carousel items from left to right.

`.carousel('pause')`

Stops the carousel from cycling through items.

`.carousel(number)`

Cycles the carousel to a particular frame (0 based, similar to an array).

`.carousel('prev')`

Cycles to the previous item.

`.carousel('next')`

Cycles to the next item.

Events

Bootstrap's carousel class exposes two events for hooking into carousel functionality.

Event	Description
slide	This event fires immediately when the <code>slide</code> instance method is invoked.
slid	This event is fired when the carousel has completed its slide transition.

Typeahead `bootstrap-typeahead.js`

Example

A basic, easily extended plugin for quickly creating elegant typeaheads with any form text input.

```
<input type="text" data-provide="typeahead">
```

Usage

Via data attributes

Add data attributes to register an element with typeahead functionality as shown in the example above.

Via JavaScript

Call the typeahead manually with:

```
$( '.typeahead' ).typeahead( )
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-source=" "`.

Name	type	default	description
source	array, function	[]	The data source to query against. May be an array of strings or a function. The function is passed two arguments, the <code>query</code> value in the input field and the <code>process</code> callback. The function may be used synchronously by returning the data source directly or asynchronously via the <code>process</code> callback's single argument.
items	number	8	The max number of items to display in the dropdown.
minLength	number	1	The minimum character length needed before triggering autocomplete suggestions
matcher	function	case insensitive	The method used to determine if a query matches an item. Accepts a single argument, the <code>item</code> against which to test the query. Access the current query with <code>this.query</code> . Return a boolean <code>true</code> if query is a match.
sorter	function	exact match, case sensitive, case insensitive	Method used to sort autocomplete results. Accepts a single argument <code>items</code> and has the scope of the typeahead instance. Reference the current query with <code>this.query</code> .
updater	function	returns selected item	The method used to return selected item. Accepts a single argument, the <code>item</code> and has the scope of the typeahead instance.
highlighter	function	highlights all default matches	Method used to highlight autocomplete results. Accepts a single argument <code>item</code> and has the scope of the typeahead instance. Should return html.

Methods

`.typeahead(options)`

Initializes an input with a typeahead.

Example

The subnavigation on the left is a live demo of the affix plugin.

Usage

Via data attributes

To easily add affix behavior to any element, just add `data-spy="affix"` to the element you want to spy on. Then use offsets to define when to toggle the pinning of an element on and off.

```
<div data-spy="affix" data-offset-top="200">...</div>
```

Heads up! You must manage the position of a pinned element and the behavior of its immediate parent. Position is controlled by `affix`, `affix-top`, and `affix-bottom`. Remember to check for a potentially collapsed parent when the affix kicks in as it's removing content from the normal flow of the page.

Via JavaScript

Call the affix plugin via JavaScript:

```
$('#navbar').affix()
```

Methods

`affix('refresh')`

When using affix in conjunction with adding or removing of elements from the DOM, you'll want to call the refresh method:

```
$('.[data-spy="affix"]').each(function () {  
    $(this).affix('refresh')  
});
```

Options

Options can be passed via data attributes or JavaScript. For data attributes, append the option name to `data-`, as in `data-offset-top="200"`.

Name	type	default	description
offset	number function object	10	Pixels to offset from screen when calculating position of scroll. If a single number is provide, the offset will be applied in both top and left directions. To listen for a single direction, or multiple unique offsets, just provided an object <code>offset: { x: 10 }</code> . Use a function when you need to dynamically provide an offset (useful for some responsive designs).

[Back to top](#)

Designed and built with all the love in the world by [@mdo](#) and [@fat](#).

Code licensed under [Apache License v2.0](#), documentation under [CC BY 3.0](#).

[Glyphicons Free](#) licensed under [CC BY 3.0](#).

- [Blog](#)
- .
- [Issues](#)
- .
- [Roadmap and changelog](#)