How it works

Before getting started with Bootstrap's modal component, be sure to read the following as our menu options have recently changed.

- Modals are built with HTML, CSS, and JavaScript. They're positioned over everything else in the document and remove scroll from the `<body>` so that modal content scrolls instead.
- Clicking on the modal "backdrop" will automatically close the modal.
- Bootstrap only supports one modal window at a time. Nested modals aren't supported as we believe them to be poor user experiences.
- Modals use `position: fixed`, which can sometimes be a bit particular about its rendering. Whenever possible, place your modal HTML in a top-level position to avoid potential interference from other elements. You'll likely run into issues when nesting a `.modal` within another fixed element.
- Once again, due to `position: fixed`, there are some caveats with using modals on mobile devices. [See our browser support docs]({{< docsref "/getting-started/browsers-devices#modals-and-dropdowns-on-mobile" >}}) for details.
- Due to how HTML5 defines its semantics, the `autofocus` HTML attribute has no effect in Bootstrap modals. To achieve the same effect, use some custom JavaScript:

```
var myModal = document.getElementById('myModal')
var myInput = document.getElementById('myInput')

myModal.addEventListener('shown.bs.modal', function () {
  myInput.focus()
})
```

{{< callout info >}} {{< partial "callout-info-prefersreducedmotion.md" >}} {{< /callout >}}

Keep reading for demos and usage guidelines.

Examples

Modal components

Below is a *static* modal example (meaning its `position` and `display` have been overridden). Included are the modal header, modal body (required for `padding`), and modal footer (optional). We ask that you include modal headers with dismiss actions whenever possible, or provide another explicit dismiss action.

Modal title

▭
Modal body text goes here.

[Close] [Save changes]

```
<div class="modal" tabindex="-1">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Modal title</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
```

```
      </div>
      <div class="modal-body">
        <p>Modal body text goes here.</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-bs-
  dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

Live demo

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

Modal title

⬭

Woohoo, you're reading this text in a modal!

| Close | Save changes |

| Launch demo modal |

```
<!-- Button trigger modal -->
<button type="button" class="btn btn-primary" data-bs-toggle="modal" data-bs-
target="#exampleModal">
  Launch demo modal
</button>

<!-- Modal -->
<div class="modal fade" id="exampleModal" tabindex="-1" aria-
labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="exampleModalLabel">Modal title</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
      </div>
      <div class="modal-body">
        ...
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div>
  </div>
</div>
```

Static backdrop

When backdrop is set to static, the modal will not close when clicking outside it. Click the button below to try it.

Modal title

I will not close if you click outside me. Don't even try to press escape key.

Close   Understood

Launch static backdrop modal

```
<!-- Button trigger modal -->
<button type="button" class="btn btn-primary" data-bs-toggle="modal" data-bs-
target="#staticBackdrop">
  Launch static backdrop modal
</button>

<!-- Modal -->
<div class="modal fade" id="staticBackdrop" data-bs-backdrop="static" data-bs-
keyboard="false" tabindex="-1" aria-labelledby="staticBackdropLabel" aria-
hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title" id="staticBackdropLabel">Modal title</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
      </div>
      <div class="modal-body">
        ...
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Understood</button>
      </div>
    </div>
  </div>
</div>
```

Scrolling long content

When modals become too long for the user's viewport or device, they scroll independent of the page itself. Try the demo below to see what we mean.

Modal title

This is some placeholder content to show the scrolling behavior for modals. Instead of repeating the text the modal, we use an inline style set a minimum height, thereby extending the length of the overall modal and demonstrating the overflow scrolling. When content becomes longer than the height of the viewport, scrolling will move the modal as needed.

Close | Save changes

Launch demo modal

You can also create a scrollable modal that allows scroll the modal body by adding `.modal-dialog-scrollable` to `.modal-dialog` .

Modal title

This is some placeholder content to show the scrolling behavior for modals. We use repeated line breaks to demonstrate how content can exceed minimum inner height, thereby showing inner scrolling. When content becomes longer than the predefined max-height of modal, content will be cropped and scrollable within the modal.

This content should appear at the bottom after you scroll.

Close Save changes

Launch demo modal

```html
<!-- Scrollable modal -->
<div class="modal-dialog modal-dialog-scrollable">
  ...
</div>
```

Vertically centered

Add `.modal-dialog-centered` to `.modal-dialog` to vertically center the modal.

Modal title

This is a vertically centered modal.

Close Save changes

Modal title

This is some placeholder content to show a vertically centered modal. We've added some extra copy here to show how vertically centering the modal works when combined with scrollable modals. We also use some repeated line breaks to quickly extend the height of the content, thereby triggering the scrolling. When content becomes longer than the predefined max-height of modal, content will be cropped and scrollable within the modal.

Just like that.

Close   Save changes

Vertically centered modal   Vertically centered scrollable modal

```
<!-- Vertically centered modal -->
<div class="modal-dialog modal-dialog-centered">
  ...
</div>

<!-- Vertically centered scrollable modal -->
<div class="modal-dialog modal-dialog-centered modal-dialog-scrollable">
  ...
</div>
```

Tooltips and popovers

[Tooltips]({{< docsref "/components/tooltips" >}}) and [popovers]({{< docsref "/components/popovers" >}}) can be placed within modals as needed. When modals are closed, any tooltips and popovers within are also automatically dismissed.

Modal title

Popover in a modal

This button triggers a popover on click.

Tooltips in a modal

This link and that link have tooltips on hover.

Close   Save changes

Launch demo modal

```
<div class="modal-body">
  <h5>Popover in a modal</h5>
  <p>This <a href="#" role="button" class="btn btn-secondary popover-test"
title="Popover title" data-bs-content="Popover body content is set in this
attribute.">button</a> triggers a popover on click.</p>
  <hr>
  <h5>Tooltips in a modal</h5>
  <p><a href="#" class="tooltip-test" title="Tooltip">This link</a> and <a href="#"
class="tooltip-test" title="Tooltip">that link</a> have tooltips on hover.</p>
</div>
```

Using the grid

Utilize the Bootstrap grid system within a modal by nesting `.container-fluid` within the `.modal-body`. Then, use the normal grid system classes as you would anywhere else.

Grids in modals

.col-md-4

.col-md-4 .ms-auto

.col-md-3 .ms-auto

.col-md-2 .ms-auto

.col-md-6 .ms-auto

Level 1: .col-sm-9

Level 2: .col-8 .col-sm-6

Level 2: .col-4 .col-sm-6

Close   Save changes

Launch demo modal

```
<div class="modal-body">
  <div class="container-fluid">
    <div class="row">
      <div class="col-md-4">.col-md-4</div>
      <div class="col-md-4 ms-auto">.col-md-4 .ms-auto</div>
    </div>
    <div class="row">
      <div class="col-md-3 ms-auto">.col-md-3 .ms-auto</div>
      <div class="col-md-2 ms-auto">.col-md-2 .ms-auto</div>
    </div>
    <div class="row">
      <div class="col-md-6 ms-auto">.col-md-6 .ms-auto</div>
    </div>
    <div class="row">
      <div class="col-sm-9">
        Level 1: .col-sm-9
        <div class="row">
          <div class="col-8 col-sm-6">
            Level 2: .col-8 .col-sm-6
          </div>
          <div class="col-4 col-sm-6">
```

```
          Level 2: .col-4 .col-sm-6
        </div>
      </div>
    </div>
  </div>
</div>
```

Varying modal content

Have a bunch of buttons that all trigger the same modal with slightly different contents? Use `event.relatedTarget` and HTML `data-bs-*` attributes to vary the contents of the modal depending on which button was clicked.

Below is a live demo followed by example HTML and JavaScript. For more information, read the modal events docs for details on `relatedTarget`.

{{< example >}} Open modal for @mdo   Open modal for @fat

Open modal for @getbootstrap

New message

Recipient:

Message:

Close   Send message

{{< /example >}}

```
var exampleModal = document.getElementById('exampleModal')
exampleModal.addEventListener('show.bs.modal', function (event) {
  // Button that triggered the modal
  var button = event.relatedTarget
  // Extract info from data-bs-* attributes
  var recipient = button.getAttribute('data-bs-whatever')
  // If necessary, you could initiate an AJAX request here
  // and then do the updating in a callback.
  //
  // Update the modal's content.
  var modalTitle = exampleModal.querySelector('.modal-title')
  var modalBodyInput = exampleModal.querySelector('.modal-body input')

  modalTitle.textContent = 'New message to ' + recipient
  modalBodyInput.value = recipient
})
```

Toggle between modals

Toggle between multiple modals with some clever placement of the `data-bs-target` and `data-bs-toggle` attributes. For example, you could toggle a password reset modal from within an already open sign in modal. **Please note multiple modals cannot be open at the same time**—this method simply toggles between two separate modals.

{{< example >}}

Modal 1

⬭

Show a second modal and hide this one with the button below.

<button>Open second modal</button>

Modal 2

⬭

Hide this modal and show the first with the button below.

<button>Back to first</button>

[Open first modal](#) {{< /example >}}

Change animation

The `$modal-fade-transform` variable determines the transform state of `.modal-dialog` before the modal fade-in animation, the `$modal-show-transform` variable determines the transform of `.modal-dialog` at the end of the modal fade-in animation.

If you want for example a zoom-in animation, you can set `$modal-fade-transform: scale(.8)`.

Remove animation

For modals that simply appear rather than fade in to view, remove the `.fade` class from your modal markup.

```html
<div class="modal" tabindex="-1" aria-labelledby="..." aria-hidden="true">
  ...
</div>
```

Dynamic heights

If the height of a modal changes while it is open, you should call `myModal.handleUpdate()` to readjust the modal's position in case a scrollbar appears.

Accessibility

Be sure to add `aria-labelledby="..."`, referencing the modal title, to `.modal`. Additionally, you may give a description of your modal dialog with `aria-describedby` on `.modal`. Note that you don't need to add `role="dialog"` since we already add it via JavaScript.

Embedding YouTube videos

Embedding YouTube videos in modals requires additional JavaScript not in Bootstrap to automatically stop playback and more. [See this helpful Stack Overflow post](#) for more information.

Optional sizes

Modals have three optional sizes, available via modifier classes to be placed on a `.modal-dialog`. These sizes kick in at certain breakpoints to avoid horizontal scrollbars on narrower viewports.

Size	Class	Modal max-width
Small	`.modal-sm`	`300px`

Default	None	`500px`
Large	`.modal-lg`	`800px`
Extra large	`.modal-xl`	`1140px`

Our default modal without modifier class constitutes the "medium" size modal.

Extra large modal Large modal Small modal

```
<div class="modal-dialog modal-xl">...</div>
<div class="modal-dialog modal-lg">...</div>
<div class="modal-dialog modal-sm">...</div>
```

Extra large modal

⬯

…

Large modal

⬯

…

Small modal

⬯

…

Fullscreen Modal

Another override is the option to pop up a modal that covers the user viewport, available via modifier classes that are placed on a `.modal-dialog`.

Class	Availability
`.modal-fullscreen`	Always
`.modal-fullscreen-sm-down`	Below `576px`
`.modal-fullscreen-md-down`	Below `768px`
`.modal-fullscreen-lg-down`	Below `992px`
`.modal-fullscreen-xl-down`	Below `1200px`
`.modal-fullscreen-xxl-down`	Below `1400px`

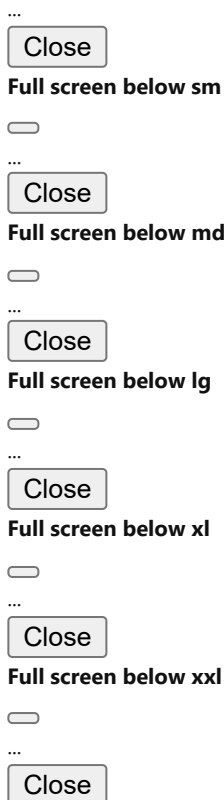Full screen Full screen below sm Full screen below md Full screen below lg
Full screen below xl Full screen below xxl

```
<!-- Full screen modal -->
<div class="modal-dialog modal-fullscreen-sm-down">
  ...
</div>
```

Full screen modal

⬯

…

Close

Full screen below sm

⬭

…

Close

Full screen below md

⬭

…

Close

Full screen below lg

⬭

…

Close

Full screen below xl

⬭

…

Close

Full screen below xxl

⬭

…

Close

Sass

Variables

{{< scss-docs name="modal-variables" file="scss/_variables.scss" >}}

Loop

[Responsive fullscreen modals](#) are generated via the `$breakpoints` map and a loop in `scss/_modal.scss`.

{{< scss-docs name="modal-fullscreen-loop" file="scss/_modal.scss" >}}

Usage

The modal plugin toggles your hidden content on demand, via data attributes or JavaScript. It also overrides default scrolling behavior and generates a `.modal-backdrop` to provide a click area for dismissing shown modals when clicking outside the modal.

Via data attributes

Toggle

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

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

Dismiss

{{% js-dismiss "modal" %}}

{{< callout warning >}} While both ways to dismiss a modal are supported, keep in mind that dismissing from outside a modal does not match [the WAI-ARIA modal dialog design pattern](#). Do this at your own risk. {{< /callout >}}

Via JavaScript

Create a modal with a single line of JavaScript:

```
var myModal = new bootstrap.Modal(document.getElementById('myModal'), options)
```

Options

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

Name	Type	Default	Description
`backdrop`	boolean or the string `'static'`	`true`	Includes a modal-backdrop element. Alternatively, specify `static` for a backdrop which doesn't close the modal on click.
`keyboard`	boolean	`true`	Closes the modal when escape key is pressed
`focus`	boolean	`true`	Puts the focus on the modal when initialized.

Methods

{{< callout danger >}} {{< partial "callout-danger-async-methods.md" >}} {{< /callout >}}

Passing options

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

```
var myModal = new bootstrap.Modal(document.getElementById('myModal'), {
  keyboard: false
})
```

toggle

Manually toggles a modal. **Returns to the caller before the modal has actually been shown or hidden** (i.e. before the `shown.bs.modal` or `hidden.bs.modal` event occurs).

```
myModal.toggle()
```

show

Manually opens a modal. **Returns to the caller before the modal has actually been shown** (i.e. before the `shown.bs.modal` event occurs).

```
myModal.show()
```

Also, you can pass a DOM element as an argument that can be received in the modal events (as the `relatedTarget` property).

```
var modalToggle = document.getElementById('toggleMyModal') // relatedTarget
myModal.show(modalToggle)
```

hide

Manually hides a modal. **Returns to the caller before the modal has actually been hidden** (i.e. before the `hidden.bs.modal` event occurs).

```
myModal.hide()
```

handleUpdate

Manually readjust the modal's position if the height of a modal changes while it is open (i.e. in case a scrollbar appears).

```
myModal.handleUpdate()
```

dispose

Destroys an element's modal. (Removes stored data on the DOM element)

```
myModal.dispose()
```

getInstance

Static method which allows you to get the modal instance associated with a DOM element

```
var myModalEl = document.getElementById('myModal')
var modal = bootstrap.Modal.getInstance(myModalEl) // Returns a Bootstrap modal
instance
```

getOrCreateInstance

Static method which allows you to get the modal instance associated with a DOM element, or create a new one in case it wasn't initialized

```
var myModalEl = document.querySelector('#myModal')
var modal = bootstrap.Modal.getOrCreateInstance(myModalEl) // Returns a Bootstrap
modal instance
```

Events

Bootstrap's modal class exposes a few events for hooking into modal functionality. All modal events are fired at the modal itself (i.e. at the `<div class="modal">`).

Event type	Description
show.bs.modal	This event fires immediately when the `show` instance method is called. If caused by a click, the clicked element is available as the `relatedTarget` property of the event.
shown.bs.modal	This event is fired when the modal has been made visible to the user (will wait for CSS transitions to complete). If caused by a click, the clicked element is available as the `relatedTarget` property of the event.
hide.bs.modal	This event is fired immediately when the `hide` instance method has been called.
hidden.bs.modal	This event is fired when the modal has finished being hidden from the user (will wait for CSS transitions to complete).
hidePrevented.bs.modal	This event is fired when the modal is shown, its backdrop is `static` and a click outside the modal or an escape key press is performed with the keyboard option or `data-bs-keyboard` set to `false`.

```javascript
var myModalEl = document.getElementById('myModal')
myModalEl.addEventListener('hidden.bs.modal', function (event) {
  // do something...
})
```