

# JavaScript

內含 13 個 jQuery 外掛，它們賦予 Bootstrap 的元件生命。非常簡單地參考全部或各別使用。

## 概觀

### 獨立或完整

jQuery 外掛都能各自獨立引用（使用 Bootstrap 獨立的 \*.js 檔案），或是一次就引用全部外掛（使用 bootstrap.js 或已最佳化的 bootstrap.min.js 檔案）。

#### 使用已編譯的 JavaScript

bootstrap.js 和 bootstrap.min.js 都內含所有外掛，僅需要引用其中一支 \*.js 即可。

#### 元件的 data- 屬性

不要在同一個元素上使用多個外掛的 data- 屬性，例如，在 button 上不能同時使用工具提示和互動視窗外掛。如果要使用多個外掛功能，那麼要使用額外容器來包裝元素。

#### 外掛相依性

某些外掛和 CSS 元件相依於其他外掛。如果你獨立引用某個外掛，請檢查文件並確保它們的相依性。注意，所有外掛都依賴於 jQuery（意思是，jQuery 必須在所有外掛檔案之前被引用）。查閱我們 bower.json (<https://github.com/twbs/bootstrap/blob/v3.3.1/bower.json>) 以瞭解那些版本的 jQuery 是被支援的。

## 譯者註

查閱 bower.json 中的 dependencies 項目。

## data- 屬性

你可以純粹透過標記 API ( data-\* 屬性 ) 就能使用所有 Bootstrap 外掛功能，並且不需要撰寫任何一行 JavaScript 程式碼。這是 Bootstrap 一流的 API，也是你在使用外掛時的首選方式。

盡管如此，在某些情況下可能很合理地需要關閉此功能。因此，我們也提供關閉 data- 屬性 API 的方法，也就是由文件命名空間 data-api 解除所有事件的繫結。看起來就像這樣：

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

或者，針對特定的外掛，僅需要在 data-api 命名空間之前加入外掛的名稱作為命名空間，像這樣：

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

## 程式 API

我們相信，你也能純粹透過 JavaScript API 使用所有的 Bootstrap 外掛。所有公開的 API 都是單獨的、可鏈結呼叫和回傳所操作的集合。

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

所有方法都接受一個選擇性的 options 物件，或一個表示特定方法的字串，或者什麼都沒有（這會初始化一個含預設行為的外掛）：

```
$('#myModal').modal() // 採用預設值初始化  
$('#myModal').modal({ keyboard: false }) // 以無鍵盤進行初始化  
$('#myModal').modal('show') // 初始化後立即呼叫 show
```

每個外掛透過 Constructor 屬性來揭露原始的建構函數：\$.fn.popover.Constructor。如果你想取得某個特定外掛的執行個體 ( instance )，可以直接從頁面元素取得： \$('[rel="popover"]').data('popover')。

## 預設設置

你可以修改外掛的預設設置，透過修改外掛的 Constructor.DEFAULTS 物件：

```
$.fn.modal.Constructor.DEFAULTS.keyboard = false // 修改互動視窗 'keyboard' 預設選項值為 false
```

## 不衝突

有時候，你需要同時使用 Bootstrap 外掛和其他 UI 框架。在這種情況下，可能偶爾會發生命名空間衝突。如果發生了這種情況，你可以呼叫外掛程式的 .noConflict 來返回原值。

```
var bootstrapButton = $.fn.button.noConflict() // 回傳 $.fn.button 的原值
$.fn.bootstrapBtn = bootstrapButton           // 取得 $(().bootstrapBtn) 的
Bootstrap 功能
```

### 譯者註

這裡的“返回原值”指的是返回一個完整功能的原始外掛物件，讓我們避開命名空間的衝突。

## Events 事件

Bootstrap 為大多數外掛的特定動作提供自訂事件。一般而言，這些事件都是以英文動詞的原型和過去分詞形式來命名呈現。動詞原型呈現在事件執行之前觸發，例如，`show`。過去分詞形式呈現在動作執行完畢之後觸發，例如，`shown`。

從 3.0.0 開始，所有 Bootstrap 事件都改為命名空間。

所有動詞原型的事件都提供 `preventDefault` 函數。這能在動作被執行之前停止執行。

```
$('#myModal').on('show.bs.modal', function (e) {
  if (!data) return e.preventDefault() // 停止互動視窗的呈現
})
```

## 當 JavaScript 被禁用時無特別補救方式

Bootstrap 的外掛在 JavaScript 被禁用情況下並無特別的補救方式。在此情況下，如果你還是很在乎使用者體驗，那麼加入 `<noscript>` (<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/noscript>) 向你的使用者說明情況（並且教導他們如果啟用 JavaScript），或者加入你自訂的補救方式說明。

### 譯者註

啟用 JavaScript 步驟（以當下主流瀏覽器版本為主）：

- IE：網際網路選項 → 安全性 → 網際網路 → 自訂等級 → 指令碼處理 → 啟用。
- Firefox：工具 → 網頁開發者 → 網頁工具箱 → 選項 → 取消勾選「停用 JavaScript」。
- Chrome：設定 → 顯示進階設定 → 隱私權 → 內容設定... → JavaScript → 選擇「允許所有網站執行 JavaScript」。

## 第三方函式庫

Bootstrap 並不提供對第三方函式庫的官方支援，例如，Prototype 或 jQuery UI 等。儘管有 `.noConflict` 和命名空間事件的處理，但還是有可能需要你自己處理相容性問題。

# 轉場效果 transition.js

## 關於轉場效果

簡單的轉場效果 ( transition )，只要在參考其他 JavaScript 檔案時一同參考 `transition.js` 檔案即可。如果你參考的是已編譯 ( 或最佳化 ) 的 `bootstrap.js` 檔案就不需要再參考此檔案 (`transition.js`)，因為它已經包含在 `bootstrap.js` 檔案之中。

## 含有什麼？

`transition.js` 是一個基本 `transitionEnd` 輔助程式以及 CSS 轉場效果模擬器。它是用來檢查其他外掛是否支援 CSS 轉場效果和擷取處理轉場效果。

### 譯者註

這從文字比較不好理解什麼是轉場效果，可參考 CSS3 transitions ([http://www.w3schools.com/css/css3\\_transitions.asp](http://www.w3schools.com/css/css3_transitions.asp))。另外，從 `transition.js` 原始碼得知，此檔案只是做瀏覽器是否支援轉場效果的檢測（由 modernizr (<http://modernizr.com/>) 提供），主要是提供給 Bootstrap 其他有使用到轉場效果的外掛使用。

換句話說，只要你有獨立使用到 Bootstrap 任何 JavaScript 外掛，建議都將 `transition.js` 包含在內。

# 互動視窗 modal.js

互動視窗 ( Modal ) 經過精簡，但更有彈性，對話方塊有著實用的功能。

### 不支援互動視窗的重疊

不要在一個已開啟的互動視窗上再開啟另一個。需要同時顯示多個互動視窗，需要自己撰寫程式碼來處理。

### 互動視窗 HTML 標記位置

總是盡量試著將互動視窗的 HTML 程式碼放置在文件最頂層的位置（也就是盡量是 `body` 的直接子元素），以避免其他元件去影響到互動視窗的展現和（或）功能。

## 行動設備注意事項

這裡有一些關於在行動設備使用互動視窗的注意事項。請查閱我們的 瀏覽器支援文件 (/bs3/GettingStarted#support-fixed-position-keyboards)。

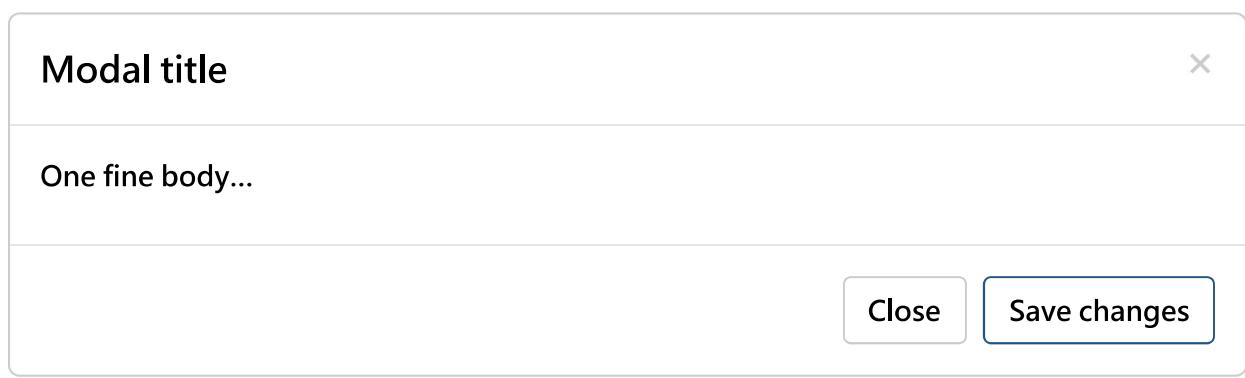
由於 HTML5 是如何定義它的語義，HTML5 `autofocus` 屬性在 Bootstrap 互動視窗並無影響。

## 範例

### 靜態範例

呈現一個帶有標題、內容以及帶有操作動作的頁尾的互動視窗。

#### EXAMPLE



```
<div class="modal fade">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal"><span aria-hidden="true">&times;</span><span class="sr-only">Close</span></button>
        <h4 class="modal-title">Modal title</h4>
      </div>
      <div class="modal-body">
        <p>One fine body&hellip;</p>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Save changes</button>
      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->
```

# 現場展示

經由點擊下面的按鈕後，會透過 JavaScript 去進行互動視窗切換。互動視窗會從頂端向下滑動且以淡入效果呈現。

## EXAMPLE

Launch demo modal

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

<!-- Modal -->
<div class="modal fade" id="myModal" tabindex="-1" role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <button type="button" class="close" data-dismiss="modal"><span aria-hidden="true">&times;</span><span class="sr-only">Close</span></button>
            <h4 class="modal-title" id="myModalLabel">Modal title</h4>
        </div>
        <div class="modal-body">
            ...
        </div>
        <div class="modal-footer">
            <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
            <button type="button" class="btn btn-primary">Save changes</button>
        </div>
    </div>
</div>
```

### 使互動視窗有更佳的無障礙

請確保將 `role="dialog"` 加入 `.modal`，`aria-labelledby="myModalLabel"` 屬性用來參考互動視窗的標題，`aria-hidden="true"` 屬性用來通知輔助技術跳過互動視窗的 DOM 元素。

另外，你應該在 `.modal` 的 `aria-describedby` 屬性描述你的互動視窗。

## 嵌入 YouTube 影片

要在 Bootstrap 中去自動停止嵌入 YouTube 影片的播放等功能，需要加入一些額外的 JavaScript，但不是在 Bootstrap 裡提供。這個 Stack Overflow (<http://stackoverflow.com/questions/18622508/bootstrap-3-and-youtube-in-modal>) 討論串有許多有用的資訊。

## 選擇性大小

互動視窗有二種可選擇大小，可經由在 `.modal-dialog` 放置修飾類別來選擇。

### EXAMPLE

[Large modal](#) [Small modal](#)

```
<!-- Large modal -->
<button type="button" class="btn btn-primary" data-toggle="modal" data-target=".bs-example-modal-lg">Large modal</button>

<div class="modal fade bs-example-modal-lg" tabindex="-1" role="dialog" aria-labelledby="myLargeModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-lg">
    <div class="modal-content">
      ...
    </div>
  </div>
</div>

<!-- Small modal -->
<button type="button" class="btn btn-primary" data-toggle="modal" data-target=".bs-example-modal-sm">Small modal</button>

<div class="modal fade bs-example-modal-sm" tabindex="-1" role="dialog" aria-labelledby="mySmallModalLabel" aria-hidden="true">
  <div class="modal-dialog modal-sm">
    <div class="modal-content">
      ...
    </div>
  </div>
</div>
```

## 移除動畫效果

如果你只想簡單的出現不想要淡入淡出的呈現效果，只需從互動視窗的標記中移除 `.fade` 類別。

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

## 基於按鈕觸發不同互動視窗內容

有一堆按鈕觸發著相同的互動視窗，但僅有少數不同的內容？使用 `event.relatedTarget` 和 `HTML data-* 屬性` ([https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Using\\_data\\_attributes](https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Using_data_attributes))（或者經由 `jQuery` (<http://api.jquery.com/data/>)）來改變決定那個按鈕被點擊時互動視窗的內容。更多 `relatedTarget` 請參考互動視窗事件說明文件。

---

### EXAMPLE

[Open modal for @mdo](#) [Open modal for @fat](#) [Open modal for @twbootstrap](#)  
...more buttons...

---

```
<button type="button" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal" data-whatever="@mdo">Open modal for @mdo</button>
<button type="button" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal" data-whatever="@fat">Open modal for @fat</button>
<button type="button" class="btn btn-primary" data-toggle="modal" data-target="#exampleModal" data-whatever="@twbootstrap">Open modal for @twbootstrap</button>
...more buttons...

<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal"><span aria-hidden="true">&times;</span><span class="sr-only">Close</span></button>
        <h4 class="modal-title" id="exampleModalLabel">New message</h4>
      </div>
      <div class="modal-body">
        <form role="form">
          <div class="form-group">
            <label for="recipient-name" class="control-label">Recipient:</label>
            <input type="text" class="form-control" id="recipient-name">
          </div>
          <div class="form-group">
            <label for="message-text" class="control-label">Message:</label>
            <textarea class="form-control" id="message-text"></textarea>
          </div>
        </form>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
        <button type="button" class="btn btn-primary">Send message</button>
      </div>
    </div>
  </div>
</div>
```

```
$('#exampleModal').on('show.bs.modal', function (event) {
  var button = $(event.relatedTarget) // Button that triggered the modal
  var recipient = button.data('whatever') // Extract info from data-
attributes
  // If necessary, you could initiate an AJAX request here (and then do the
updating in a callback).
  // Update the modal's content. We'll use jQuery here, but you could use a
data binding library or other methods instead.
  var modal = $(this)
  modal.find('.modal-title').text('New message to ' + recipient)
  modal.find('.modal-body input').val(recipient)
})
```

# 使用方法

經由 data- 屬性或 JavaScript，可以依照需求去呼叫互動視窗外掛切換隱藏的內容。在顯示時，它還會加入 .modal-open 到 <body> 元素以覆寫預設的滑動行為，並且產生一個 .modal-backdrop 以提供一個可點擊的區域，當在互動視窗之外產生點擊行為時會關閉顯示的互動視窗。

## 經由 data- 屬性

要啟用一個互動視窗不需要寫 JavaScript。在要控制的元素上設置 data-toggle="modal"，例如，按鈕元素，接著用 data-target="#foo" 或 href="#foo" 指定到進行切換為互動視窗的目標元素上。

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

## 經由 JavaScript

僅需一行 JavaScript，就能透過元素 id（例如：myModal）進行呼叫：

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

## 選項

選項可以經由 data- 屬性或 JavaScript 來傳遞。使用 data- 屬性，附加選項的名稱到 data- 之後，例如，data-backdrop=""。

名稱	類型	預設	說明
backdrop	boolean or the string 'static'	true	包含 modal-backdrop 元素。另外，指定 static 的 backdrop，在互動視窗上點擊不會關閉。
keyboard	boolean	true	當鍵盤 Esc 按鈕被按下關閉互動視窗。
show	boolean	true	當初始化完成呈現互動視窗。Shows the modal when initialized.

名稱	類型	預設	說明
remote	path	false	<p>此選項從 v3.3.0 開始不再推薦並且在 v4 將被移除。我們推薦替代性使用用戶端範例或資料繫結 ( data binding ) 框架，或自己使用 jQuery.load (<a href="http://api.jquery.com/load/">http://api.jquery.com/load/</a>) 來呼叫。</p> <p>如果有提供遠端 URL，經由 jQuery 的 load 方法內容將會同時被載入和注入到 .modal-body 類別的 div 元素中。如果你是使用 data-api 的話，你可能是使用 href 標記來指定遠端資源。例如以下範例所示：</p> <pre>&lt;a data-toggle="modal" href="remote.html" data-target="#modal"&gt;Click me&lt;/a&gt;</pre>

## 方法

### .modal(options)

將你指定的內容轉換為互動視窗。接受一個選擇性的 object 選項。

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

### .modal('toggle')

手動進行顯示或隱藏互動視窗的切換。在互動視窗顯示或隱藏之前返回呼叫者 ( caller ) ( 也就是 shown.bs.modal 或 hidden.bs.modal 事件發生之前 ) 。

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

### .modal('show')

手動顯示互動視窗。在互動視窗顯示之前返回呼叫者 ( caller ) ( 也就是 shown.bs.modal 事件發生之前 ) 。

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

## .modal('hide')

手動藏隱互動視窗。在互動視窗隱藏之前返回呼叫者 ( caller ) ( 也就是 hidden.bs.modal 事件發生之前 ) 。

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

## Events

Bootstrap 的互動視窗公開了一些事件，用於監聽這些事件並可以執行自己的程式碼。

事件類別	說明
show.bs.modal	當 show 執行個體方法被呼叫，此事件會立即被觸發。如果觸發原因是某個元素被點擊，則被點擊的元素可透過事件的 relatedTarget 屬性來存取。
shown.bs.modal	當互動視窗已經在使用者面前顯示完成（且 CSS 轉場效果已經完成），此事件會立即被觸發。如果觸發原因是某個元素被點擊，則被點擊的元素可透過事件的 relatedTarget 屬性來存取。
hide.bs.modal	當 hide 執行個體方法被呼叫，此事件會立即被觸發。
hidden.bs.modal	當互動視窗已經在使用者面前隱藏完成（且 CSS 轉場效果已經完成），此事件會立即被觸發。
loaded.bs.modal	當互動視窗使用 remote 資源選項來載入內容，載入完成後，此事件會立即被觸發。

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

## 下拉選單 dropdown.js

### 範例

使用這個簡單的外掛，幾乎所有內容都能加入下拉選單，包含巡覽列、頁籤、藥片樣式。

### 在巡覽列之內

#### EXAMPLE

# 在藥片樣式之內

## EXAMPLE

Regular link    Dropdown ▾    Dropdown ▾    Dropdown ▾

## 使用方法

經由 `data-` 屬性或 JavaScript。下拉選單外掛在父清單項目上切換 `.open` 類別，以切換隱藏的選單內容。

在行動設備上，為開啟的下拉選單加入 `.dropdown-backdrop` 作為一個點選區域，當點選至選單區域之外時可以觸發下拉選單的關閉，這需要適當的 iOS 支援。這意味著，在行動設備上，從開啟的下拉選單要切換到另一個不同的下拉選單需要一個額外的點選區域。

注意：`data-toggle="dropdown"` 屬性屬於應用程式層級的關閉下拉選單，這是一個好主意，我們建議始終使用它。

## 經由 `data-` 屬性

加入 `data-toggle="dropdown"` 到連結或按鈕，以切換下拉選單。

```
<div class="dropdown">
  <button id="dLabel" type="button" data-toggle="dropdown" aria-
haspopup="true" role="button" aria-expanded="false">
    Dropdown trigger
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
    ...
  </ul>
</div>
```

為完整保留 URL，請使用 `data-target` 屬性替代 `href="#"`。

```
<div class="dropdown">
  <a id="dLabel" data-target="#" href="http://example.com" data-
toggle="dropdown" aria-haspopup="true" role="button" aria-expanded="false">
    Dropdown trigger
    <span class="caret"></span>
  </a>

  <ul class="dropdown-menu" role="menu" aria-labelledby="dLabel">
    ...
  </ul>
</div>
```

# 經由 JavaScript

透過 JavaScript 呼叫下拉式選單：

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

**仍然需要 `data-toggle="dropdown"`**

無論你是使用 JavaScript 或改用 data-api 來呼叫下拉選單，`data-toggle="dropdown"` 總是需要設置在觸發下拉選單元素的父元素上。

## 選項

無

## 方法

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

切換巡覽列或頁籤巡覽的下拉選單。

## 事件

所有下拉選單事件都會在 `.dropdown-menu` 的父元素被觸發。

事件型別	說明
<code>show.bs.dropdown</code>	當顯示執行體個方法被呼叫，此事件會立即被觸發。切換的連結元素可作為事件的 <code>relatedTarget</code> 屬性來使用。
<code>shown.bs.dropdown</code>	當下拉選單已經在使用者面前顯示完成（且 CSS 轉場效果已經完成），此事件會立即被觸發。切換的連結元素可作為事件的 <code>relatedTarget</code> 屬性來使用。
<code>hide.bs.dropdown</code>	當隱藏執行體個方法被呼叫，此事件會立即被觸發。切換的連結元素可作為事件的 <code>relatedTarget</code> 屬性來使用。
<code>hidden.bs.dropdown</code>	當下拉選單已經在使用者面前隱藏完成（且 CSS 轉場效果已經完成），此事件會立即被觸發。切換的連結元素可作為事件的 <code>relatedTarget</code> 屬性來使用。

```
$('#myDropdown').on('show.bs.dropdown', function () {  
    // do something...  
})
```

# 滑動監視 scrollspy.js

## 巡覽列範例

滑動監視外掛會自動依據滑動位置來更新巡覽列裡的目標。滑動下面區域使其低於巡覽列，然後觀察 active 類別帶來的變化。下拉式選單的子項目會跟著變化為醒目提示效果。

### EXAMPLE

@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

### 譯者註

.active 類別的變化必須使用開發者工具來觀察。當下拉滑動接觸到內容的標題（@fat 或 @mdo）時，滑動監視外掛會動態為對應巡覽列裡的 <li> 加入 active 類別，以呈現醒目提示效果。

### 譯者註

此範例在 Windows App 模式下並無互動效果。

## 使用方法

### 需要巡覽元件

滑動監視目前需要使用到巡覽元件 (/bs3/Components#nav) 以提供正確的醒目提示與 active 連結。

### 必須可解析 ID 目標

巡覽連結必須是可解析 id 目標。例如，<a href="#home">home</a> 必須在 DOM 中有對應元素，像是 <div id="home"></div>。

## 不可見 ( :visible ) 目標元素會忽略

目標元素如果在 jQuery 不是可見的 ( :visible (<http://api.jquery.com/visible-selector/>) )，那麼根據 jQuery 的行為將會被忽略，對應的巡覽將永遠不會有醒目提示的效果。

## 需要相對定位

無論執行方法，滑動監視需要在你監視的元素上使用 position: relative;，通常是在 <body>。

### 經由 data- 屬性

相當容易就能加入滑動監視的行為到你的頂層巡覽列，加入 data-spy="scroll" 到你想監視的元素上（大部分情況是加到 <body> 上），然後加入 data-target 屬性與 CSS 的 id 或 class 名稱到任何 Bootstrap .nav 元件的父元素上。

```
body {  
    position: relative;  
}
```

```
<body data-spy="scroll" data-target=".navbar-example">  
    ...  
    <div class="navbar-example">  
        <ul class="nav nav-tabs" role="tablist">  
            ...  
        </ul>  
    </div>  
    ...  
</body>
```

### 經由 JavaScript

在加入 position: relative; 之後，就能透過 JavaScript 呼叫滑動監視行為：

```
$( 'body' ).scrollspy({ target: '.navbar-example' })
```

## 方法

### .scrollspy('refresh')

當使用滑動監視的 DOM 元素有新增或移除操作時，你需要去呼叫 refresh 方法，如下所示：

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

## 選項

選項能透過 `data-` 屬性或 JavaScript 來傳遞。使用 `data-` 屬性傳遞，將選項名稱附加至 `data-` 後面，例如，`data-offset=""`。

名稱	類型	預設	說明
offset	number	10	計算滑動位置時從頂端要位移的像素。

## 事件

類型	說明
activate.bs.scrollspy	當一個項目被滑動監視設置啟用時，此事件會被觸發。

```
$('#myScrollspy').on('activate.bs.scrollspy', function () {  
    // do something...  
})
```

## 可切換頁籤 tab.js

### 頁籤範例

快速加入動態頁籤 ( tab ) 功能到一個轉換用的頁面以切換本地內容，甚至經由下拉式選單。

#### EXAMPLE

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.

#### 擴充頁籤的巡覽

此外掛擴充頁籤元件 ([/bs3/Components#nav-tabs](#)) 巡覽功能到可切換頁籤區域。

# 使用方法

啟用切換頁籤經由 JavaScript ( 每一個頁籤需要單獨啟用 ) :

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

你有數種方式來單獨啟用頁籤 :

```
$('#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)
```

## 標記

僅需在元素上指定 `data-toggle="tab"` 或 `data-toggle="pill"` 即可啟用頁籤或藥片樣式巡覽而不必撰寫任何 JavaScript。在頁籤的 `ul` 設置 `nav` 與 `nav-tabs` 類別即可套用 Bootstrap 頁籤元件 (/bs3/Components#nav-tabs)，而加入 `nav` 和 `nav-pills` 類別將會套用藥片樣式 (/bs3/Components#nav-pills)。

```
<div role="tabpanel">  
  
    <!-- Nav tabs -->  
    <ul class="nav nav-tabs" role="tablist">  
        <li role="presentation" class="active"><a href="#home" aria-  
            controls="home" role="tab" data-toggle="tab">Home</a></li>  
        <li role="presentation"><a href="#profile" aria-controls="profile"  
            role="tab" data-toggle="tab">Profile</a></li>  
        <li role="presentation"><a href="#messages" aria-controls="messages"  
            role="tab" data-toggle="tab">Messages</a></li>  
        <li role="presentation"><a href="#settings" aria-controls="settings"  
            role="tab" data-toggle="tab">Settings</a></li>  
    </ul>  
  
    <!-- Tab panes -->  
    <div class="tab-content">  
        <div role="tabpanel" class="tab-pane active" id="home">...</div>  
        <div role="tabpanel" class="tab-pane" id="profile">...</div>  
        <div role="tabpanel" class="tab-pane" id="messages">...</div>  
        <div role="tabpanel" class="tab-pane" id="settings">...</div>  
    </div>  
  
</div>
```

## 淡入效果

要讓頁籤有淡入效果，需要在每個 `.tab-pane` 加入 `.fade` 類別。而且第一個頁籤頁面需要加入 `.in`

類別，以正確初始化淡入效果內容。

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

## 方法

### `$(().tab`

啟用一個頁籤元素和內容容器。頁籤應該包含 `data-target` 或 `href` 屬性且指向 DOM 中某個容器節點。

```
<ul class="nav nav-tabs" role="tablist" id="myTab">
  <li role="presentation" class="active"><a href="#home" aria-controls="home" role="tab" data-toggle="tab">Home</a></li>
  <li role="presentation"><a href="#profile" aria-controls="profile" role="tab" data-toggle="tab">Profile</a></li>
  <li role="presentation"><a href="#messages" aria-controls="messages" role="tab" data-toggle="tab">Messages</a></li>
  <li role="presentation"><a href="#settings" aria-controls="settings" role="tab" data-toggle="tab">Settings</a></li>
</ul>

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

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

### 譯者註

注意範例中每個 `href="#idname"` 與 `<div id="name">` 的對應關係。也就是 HTML 的锚點關係。

## 事件

當顯示一個新頁籤，事件會按以下順序觸發：

1. `hide.bs.tab` ( 在當下啟用的頁籤 )

2. show.bs.tab (在將被顯示的頁籤)
3. hidden.bs.tab (在前一個啟用的頁籤，和 show.bs.tab 事件相同作用)
4. shown.bs.tab (在新啟用顯示的頁籤，和 show.bs.tab 事件相同作用)

如果沒有頁籤被啟用，那麼 hide.bs.tab 和 hidden.bs.tab 事件將不會被觸發。

類型	說明
show.bs.tab	此事件在頁籤呈現時觸發，但是是在新頁籤呈現完成之前。使用 event.target 指向要啟用的頁籤，使用 event.relatedTarget 指向前一個啟用的頁籤（如果有的話）。
shown.bs.tab	此事件在頁籤已經完成呈現之後觸發。使用 event.target 指向要啟用的頁籤，使用 event.relatedTarget 指向前一個啟用的頁籤（如果有的話）。
hide.bs.tab	當一個新頁籤將被顯示時，此事件會被觸發（因此，前一個啟用的頁籤是被隱藏的）。分別使用 event.target 指向當下啟用的頁籤，使用 event.relatedTarget 指向即將啟用的新頁籤。
hidden.bs.tab	當一個新頁籤將被顯示之後，此事件會被觸發（因此，前一個啟用的頁籤是被隱藏的）。分別使用 event.target 指向前一個啟用的頁籤，使用 event.relatedTarget 指向新啟用的頁籤。

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

## 提示訊息 tooltip.js

取自 Jason Frame 所撰寫優秀 jQuery.tipsy 外掛得到的靈感；提示訊息是一個更新版本，它不依賴圖片，改用 CSS3 的動畫和 data- 屬性來對標題進行本地標題儲存。

### 譯者註

本地標題儲存意思是，將提示的描述（或說明）放入元素的 title 屬性內。

長度為零的標題，提示訊息將會不顯示。

## 範例

將滑鼠移至連結上將可看到提示訊息：

---

## EXAMPLE

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.

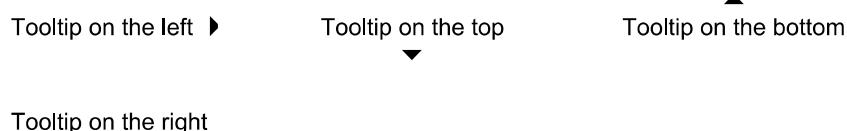
---

## 靜態提示訊息

有四個可用的選項：top、right、bottom 和 left 對齊方式。

---

## EXAMPLE



---

## 四種方向

---

## EXAMPLE

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

---

```
<button type="button" class="btn btn-default" data-toggle="tooltip" data-placement="left" title="Tooltip on left">Tooltip on left</button>
```

```
<button type="button" class="btn btn-default" data-toggle="tooltip" data-placement="top" title="Tooltip on top">Tooltip on top</button>
```

```
<button type="button" class="btn btn-default" data-toggle="tooltip" data-placement="bottom" title="Tooltip on bottom">Tooltip on bottom</button>
```

```
<button type="button" class="btn btn-default" data-toggle="tooltip" data-placement="right" title="Tooltip on right">Tooltip on right</button>
```

---

### 宣告目的

出於性能原因，提訊息訊和彈出提示的 data-api 必須明確宣告，意思是，你必須自行初始化它們。

有一種方式來初始化頁面上所有提示訊息，透過 data-toggle 屬性來選擇它們自己。

```
$(function () {
  $('[data-toggle="tooltip"]').tooltip()
})
```

### 提示訊息在按鈕群組與 input 群組需要特別設置

當你在 .btn-group 或 .input-group 裡使用提示訊息，你必須設置 container: 'body' (文件後面說明) 以避免不必要的副作用 (例如，工具提示被觸發時，會讓元素變寬和 (或) 失去圓角效果)。

### 不要嘗試在隱藏元素使用提示訊息

當呼叫 `$(...).tooltip('show')` 的目標元素是 `display: none;` 狀態，那麼會造成提示訊息被不正常定位的情況。

### 在被禁用的元素上使用提示訊息需要額外包裝的容器

加入提示訊息到被套用 `disabled` 或 `.disabled` 的元素上，請將元素放置到 `<div>` 內並且將提示訊息套用到 `<div>` 上。

## 使用方法

提示訊息外掛依照需求來產生內容的標記，並預設情況下，放置提示訊息之後會觸發它們的元素。

經由 JavaScript 觸發提示訊息：

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

## 標記

在你希望提供提示訊息的 HTML 元素，僅僅需要設 `data-` 屬性和 `title` 屬性。產生標記的提示訊息是相當簡單，但它需要一個明確的位置 (預設外掛是設置為 `top` 方向)。

### 多行的連結

有時你會想要把提示訊息加到連結裡，但此連結包裝了多行資訊，提示訊息外掛的預設行為會置中對齊，不論是水平或垂直。加入 `white-space: nowrap;` 到你的連結，可以避免此問題。

```

<!-- HTML to write -->
<a href="#" data-toggle="tooltip" title="Some tooltip text!">Hover over me</a>

<!-- Generated markup by the plugin -->
<div class="tooltip top" role="tooltip">
  <div class="tooltip-arrow"></div>
  <div class="tooltip-inner">
    Some tooltip text!
  </div>
</div>

```

## 選項

選項能透過 `data-` 屬性或 JavaScript 來傳遞。使用 `data-` 屬性傳遞，將選項名稱附加至 `data-` 後面，例如，`data-animation=""`。

名稱	類型	預設	說明
<code>animation</code>	<code>boolean</code>	<code>true</code>	為提示訊息加入一個 CSS 淡出入轉換的效果。
<code>container</code>	<code>string   false</code>	<code>false</code>	附加提示訊息到指定的元素。例如， <code>container: 'body'</code> 。此選項在提示訊息需要在一個流動的近觸發元素要定位時特別有用，這可以預防提示窗調整大小的從觸發的元素中流走。
<code>delay</code>	<code>number   object</code>	<code>0</code>	<p>呈現或隱藏提示訊息的延遲時間（毫秒），注意設置到手動（<code>manual</code>）類型觸發器上。</p> <p>如果提供的數字合法，延遲設置會同時套用到所有顯示上。</p> <p>物件建構式：<code>delay: { "show": 500, "hide": 100 }</code></p>
<code>html</code>	<code>boolean</code>	<code>false</code>	插入 HTML 元素到提示訊息。如果為 <code>false</code> ，則使用 <code>text</code> 方法來插入內容到 DOM 中。使用此選項時，你必須小心關於 XSS (Cross-site scripting) 險。

名稱	類型	預設	說明
placement	string   function	'top'	<p>如何定位提示訊息的方向 - top   bottom   left   auto。</p> <p>當指定為 "auto" 時，會動態調整提示訊息的方向。例如，當 placement 指定為 "auto left" 時，提示訊息會盡可能的在左邊呈現，否則會在右邊呈現。</p> <p>當函式決定使用 placement，它會經由第一個參數來指出提示訊息的 DOM (Document Object Model) 節點，並且經由第二個參數來指出要觸發 DOM (Document Object Model) 節點的動作。tool 裡的 this 設置為提示訊息的執行個體。</p>
selector	string	false	<p>如果有提供 selector，提示訊息物件將會被委派到合乎條件的目標上。在實務上，這會使用在動的 HTML 內容去加入提示訊息功能。可以到 <a href="#">Bootstrap Issues 4215</a> (<a href="https://github.com/twbs/bootstrap/issues/4215">https://github.com/twbs/bootstrap/issues/4215</a>) 和 selector 範例 (<a href="http://jsbin.com/zopod/1/">http://jsbin.com/zopod/1/</a>) 解更多資訊。</p>
template	string	'<div class="tooltip" role="tooltip"><div class="tooltip-arrow"></div><div class="tooltip-inner"></div></div>'	<p>當建立提示訊息時要使用的範本 HTML。</p> <p>提示訊息的 title 將會被注入到 .tooltip-in 裡。</p> <p>.tooltip-arrow 會成為提示訊息指示方向。</p> <p>最外層的元素應該含有 .tooltip 類別。</p>
title	string   function	" "	<p>如果 title 屬性不存在，使用此值做為預設標記。</p> <p>如果是一個 function 被賦予，它將會呼叫 this 考並設置提示訊息附加到元素上。</p>
trigger	string	'hover focus'	提示訊息的觸發方式 - click   hover   focus   manual。你可以傳遞多個觸發器，使用空格分開他們。
viewport	string   object	{ selector: 'body', padding: 0 }	保持此元素邊界內的提示訊息。範例：viewport: '#viewport' 或 { "selector": "#viewport", "padding": 0 }



## 譯者註

click 是點擊時觸發； hover 是滑鼠移入時觸發； focus 是取得焦點時觸發； manual 是手動觸發。

動控制觸發。

## 各別提示訊息的 data- 屬性

如前所說，各別提示訊息的選項也可以透過 data- 屬性來指定。

# 方法

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

附加提示訊息處理常式到元素的集合中。

## `.tooltip('show')`

呈現元素的提示訊息。title 屬性長度為零的提示訊息將不會呈現。

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

## `.tooltip('hide')`

隱藏元素的提示訊息。

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

## `.tooltip('toggle')`

進行元素提示訊息 show 或 hide 狀態的切換

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

## `.tooltip('destroy')`

隱藏和銷毀元素的提示訊息。

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

# 事件

類型	說明
<code>show.bs.tooltip</code>	當 show 執行個體方法被呼叫，此事件會立即被觸發。
<code>shown.bs.tooltip</code>	當提示訊息已經在使用者面前顯示完成（且 CSS 轉場效果已經完成），此事件會被觸發。
<code>hide.bs.tooltip</code>	當 hide 執行個體方法被呼叫，此事件會立即被觸發。
<code>hidden.bs.tooltip</code>	當提示訊息已經在使用者面前隱藏完成（且 CSS 轉場效果已經完成），此事件會被觸發。

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

# 彈出提示 popover.js

與那些 iPad 類似的效果，為任意元素加入一個小覆蓋層以加入額外資訊。

彈出提示在標題與內容長度為零時不會顯示。

## 外掛相依性

彈出提示依賴提示訊息外掛，請包含在你使用的 Bootstrap 版本中。

## 宣告目的

出於性能原因，提訊息和彈出提示的 data-api 必須明確宣告，意思是，你必須自行初始化它們。

有一種方式來初始化頁面上所有彈出提示，透過 data-toggle 屬性來選擇它們自己。

```
$(function () {  
    $('[data-toggle="popover"]').popover()  
})
```

## 彈出提示在按鈕群組與 input 群組需要特別設置

當你在 .btn-group 或 .input-group 裡使用彈出提示，你必須設置 container: 'body' (文件後面說明) 以避免不必要的副作用 (例如，彈出提示被觸發時，會讓元素變寬和 (或) 失去圓角效果)。

## 不要嘗試在隱藏元素使用彈出提示

當呼叫 \$(...).popover('show') 的目標元素是 display: none; 狀態，那麼會造成提示訊息被不正常定位的情況。

## 在被禁用的元素上使用彈出提示需要額外包裝的容器

加入彈出提示到被套用 disabled 或 .disabled 的元素上，請將元素放置到 <div> 內並且將

彈出提示套用到 `<div>` 上。

## 多行的連結

有時你會想要把彈出提示加到連結裡，但此連結包裝了多行資訊，彈出提示外掛的預設行為會置中對齊，不論是水平或垂直。加入 `white-space: nowrap;` 到你的連結，可以避免此問題。

# 範例

## 靜態彈出提示

有四個方向可選擇：`top`、`right`、`bottom` 和 `left` 來對齊。

### EXAMPLE

#### 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.

# 現場展示

### EXAMPLE

點擊切換彈出提示

```
<button type="button" class="btn btn-lg btn-danger" data-toggle="popover" title="Popover title" data-content="And here's some amazing content. It's very engaging. Right?">點擊切換彈出提示</button>
```

## 四個方向

### EXAMPLE

向左彈出 向上彈出 向下彈出 向右彈出

```
<button type="button" class="btn btn-default" data-container="body" data-toggle="popover" data-placement="left" data-content="Vivamus sagittis lacus vel augue laoreet rutrum faucibus.">  
    向左彈出  
</button>
```

```
<button type="button" class="btn btn-default" data-container="body" data-toggle="popover" data-placement="top" data-content="Vivamus sagittis lacus vel augue laoreet rutrum faucibus.">  
    向上彈出  
</button>
```

```
<button type="button" class="btn btn-default" data-container="body" data-toggle="popover" data-placement="bottom" data-content="Vivamus sagittis lacus vel augue laoreet rutrum faucibus.">  
    向下彈出  
</button>
```

```
<button type="button" class="btn btn-default" data-container="body" data-toggle="popover" data-placement="right" data-content="Vivamus sagittis lacus vel augue laoreet rutrum faucibus.">  
    向右彈出  
</button>
```

## 下次點擊時取消解除狀態

在使用者下一次點擊時，使用 `focus` 來觸發取消解除彈出提示的功能。

### 使用取消解除狀態時，需指定特定的標記

為了適當的跨瀏覽器與跨平台行為，你應當使用 `<a>` 標籤，而不是使用 `<button>` 標籤，並且你必須含有 `tabindex` ([https://developer.mozilla.org/en-US/docs/Web/HTML/Global\\_attributes#tabindex](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes#tabindex)) 屬性。

---

## EXAMPLE

取消解除狀態

```
<a href="#" tabindex="0" class="btn btn-lg btn-danger" role="button" data-toggle="popover" data-trigger="focus" title="Dismissible popover" data-content="And here's some amazing content. It's very engaging. Right?">取消解除狀態</a>
```

## 使用方法

經由 JavaScript 啟用彈出提示：

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

## 選項

選項能透過 data- 屬性或 JavaScript 來傳遞。使用 data- 屬性傳遞，將選項名稱附加至 data- 後面，例如，`data-animation=""`。

名稱	類型	預設	說明
animation	boolean	true	為彈出提示加入一個 CSS 淡入轉換的效果
container	string   false	false	附加彈出提示到指定的元素，例如： <code>container: 'body'</code> 。此選項在下列情況特別有用：它允許彈出提示，在浮動的文件附近觸發元素 - 這意味著提示在調整視窗大小的觸發元素中漂流走。
content	string   function	" "	如果 <code>data-content</code> 屬性不存在，使用此值作為內容。 如果一個 <code>function</code> 被指定，它將會呼叫它並參考設定到元素上，彈出提示會被附加上去。
delay	number   object	0	呈現或隱藏彈出提示的延遲時間（毫秒），適用於自動觸發器類型。 如果提供合法的數字， <code>delay</code> 設置會同時套用到 <code>show</code> 效果上。 物件建構式： <code>delay: { "show": 500, "hide": 100 }</code>

名稱	類型	預設	說明
html	boolean	false	插入 HTML 到彈出提示。如果為 false，jQuery 方法來插入內容到 DOM (Document Model) 之中。使用 text 方法，你必須關心 (Cross-site scripting) 功擊的防禦。
placement	string   function	'right'	<p>如何去定位彈出提示的位置 - top   bottom right。</p> <p>當指定為 " auto "，它會動態調整彈出提示的如，如果 placement 設置為 " auto left 的提示會盡可能出現在左邊，否則出現在右邊。</p> <p>當函式決定使用 placement，它會經由第一指出彈出提示的 DOM 節點，並且經由第二指出要觸發 DOM 節點的動作。popover.js 設置為彈出提示的執行個體。</p>
selector	string	false	<p>如果有提供 selector，彈出提示物件將會結合條件的目標上。在實務上，這會使用在動態內容套件彈出提示。進行細節可以查看 Bootstrap Issue 4215 (<a href="https://github.com/twbs/bootstrap/issues/4215">https://github.com/twbs/bootstrap/issues/4215</a>) 和一個範例 (<a href="http://jsbin.com/zopod/1/edit?html,js">http://jsbin.com/zopod/1/edit?html,js</a>)</p>
template	string	'<div class="popover" role="tooltip"><div class="arrow"></div><h3 class="popover-title"></h3><div class="popover-content"></div></div>'	<p>基礎 HTML，用來建立彈出提示時使用。</p> <p>彈出提示的 title 會被注入到 .popover-title 中。</p> <p>彈出提示的 content 會被注入到 .popover-content 中。</p> <p>.arrow 會變成彈出提示的箭頭。</p> <p>最外層包覆的元素應該含有 .popover 類別</p>
title	string   function	" "	<p>如果 title 屬性不存在，使用此值做為預設</p> <p>如果一個 function 被指定，它將會呼叫它的考設定到元素上，彈出提示會被附加上去。</p>
trigger	string	'click'	彈出提示觸發方式 - click   hover   focus manual。你也可以傳遞多個觸發器，使用空們。
viewport	string   object	{ selector: 'body', padding: 0 }	保持此彈出提示在元素的邊界之內。例如： '#viewport' 或 { "selector": "#viewport", "padding": 0 }

## 個別彈出提示的 data- 屬性

對於個別的彈出提示選項也可以透過 data- 屬性來指定，如上述解釋。

## Methods方法

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

在一組元素集合上初始化彈出提示。

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

呈現元素的彈出提示。 title 屬性和內容長度為零的彈出提示將不會呈現。 Reveals an element's popover. Popovers whose both title and content are zero-length are never displayed.

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

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

隱藏元素的彈出提示。

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

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

切換彈出提示的狀態（呈現/隱藏）。

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

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

隱藏與刪除元素的彈出提示。

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

## 事件

事件類別	說明
<code>show.bs.popover</code>	當 show 執行個體方法被呼叫，此事件會立即被觸發。
<code>shown.bs.popover</code>	當彈出提示已經在使用者面前顯示完成（且 CSS 轉場效果已經完成），此事件會被觸發。
<code>hide.bs.popover</code>	當 hide 執行個體方法被呼叫，此事件會立即被觸發。
<code>hidden.bs.popover</code>	當彈出提示已經在使用者面前隱藏完成（且 CSS 轉場效果已經完成），此事件會被觸發。

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

## 警告訊息 alert.js

### 警告範例

利用此外掛對所有警告訊息加入取消功能。

當按鈕使用 `.close` 按鈕，它必須是 `.alert-dismissible` 之後第一個子元素，才能沒有任何文字內容在標記之中。

#### EXAMPLE

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](#)

#### 譯者註

取消功能是指最右邊的「X」。

### 使用方法

僅需要加入 `data-dismiss="alert"` 到你的關閉按鈕，即可自動賦予警告訊息閉關的功能。關閉警告訊息將會它從 DOM 之中移除。

```
<button type="button" class="close" data-dismiss="alert">  
  <span aria-hidden="true">&times;</span>  
  <span class="sr-only">Close</span>  
</button>
```

為了關閉警告時能有動畫效果，確定它們有套用到 `.fade` 和 `.in` 類別。

## 方法

### `$(().alert()`

使警告訊息能監聽 `data-dismiss="alert"` 屬性的後代元素的 `click` 事件。（當使用 `data-` API 自動初始化時不需要）。

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

關閉警告訊息並從 DOM 移除它。如果 `.fade` 和 `.in` 類別存在在元素上，那麼警告元素會在淡出效果之後才會移除。

## 事件

Bootstrap 的警告訊息類別有公開一些事件允許監聽警告功能。

事件類別	說明
<code>close.bs.alert</code>	當 <code>close</code> 執行個體方法被呼叫，此事件會立即被觸發。
<code>closed.bs.alert</code>	當警告訊息已經在使用者面前關閉完成（且 CSS 轉場效果已經完成），此事件會被觸發。

```
$('#myAlert').on('closed.bs.alert', function () {  
  // do something...  
})
```

## 按鈕 button.js

讓按鈕可以做更多事。控制按鈕的狀態或建立複合元件（像是工具列）的按鈕群組。

### 跨瀏覽器相容性

Firefox 跨頁載入時保持禁用的狀態 (<https://github.com/twbs/bootstrap/issues/793>)。變通方案是在按鈕上使用 `autocomplete="off"`。細節可查看 Mozilla bug #654072 ([https://bugzilla.mozilla.org/show\\_bug.cgi?id=654072](https://bugzilla.mozilla.org/show_bug.cgi?id=654072))。

## 狀態

加入 `data-loading-text="Loading..."` 使按鈕呈現載入中的狀態。

## 無論你喜歡那種狀態！

以下這個展示範例，我們使用 `data-loading-text` 和 `$(().button('loading'))`，但那不是你  
能使用狀態的唯一方式。閱讀後面關於 `$(().button(string))` 文件以瞭解更多。

### EXAMPLE

#### Loading state

```
<button type="button" id="myButton" data-loading-text="Loading..." class="btn  
btn-primary" autocomplete="off">  
    Loading state  
</button>  
  
<script>  
    $('#myButton').on('click', function () {  
        var $btn = $(this).button('loading')  
        // business logic...  
        $btn.button('reset')  
    })  
</script>
```

## 單一切換

加入 `data-toggle="button"` 啟用單一按鈕的切換。

預切換按鈕需要 `.active` 和 `aria-pressed="true"`

預切換的按鈕，你必須自己加入 `.active` 類別和 `aria-pressed="true"` 屬性到 `button` 裡。

### EXAMPLE

#### Single toggle

```
<button type="button" class="btn btn-primary" data-toggle="button" aria-  
pressed="false" autocomplete="off">  
    Single toggle  
</button>
```

#### 譯者註

想成一個開關，按第一下開、按第二下關。

# Checkbox 與 Radio

將 `data-toggle="buttons"` 加入到包含 checkbox 或 radio 的 `.btn-group` 裡，它們會切換各自的樣式。

## 預勾選選項需要 `.active`

對於預先勾選選項，你必須自己加入 `.active` 類別到 `label` 元素中。

## 視覺上的選取狀態，僅有在點擊時會更新

如果一個 checkbox 按鈕的選取狀態在按鈕上沒有觸發 `click` 事件（例如，經由 `<input type="reset">` 或經由設定 `input` 元素的 `checkbox` 屬性），你必須手動在 `input` 元素 `label` 切換 `.active` 類別。

## EXAMPLE

Checkbox 1 (pre-checked)	Checkbox 2	Checkbox 3
--------------------------	------------	------------

```
<div class="btn-group" data-toggle="buttons">
  <label class="btn btn-primary active">
    <input type="checkbox" autocomplete="off" checked> Checkbox 1 (pre-checked)
  </label>
  <label class="btn btn-primary">
    <input type="checkbox" autocomplete="off"> Checkbox 2
  </label>
  <label class="btn btn-primary">
    <input type="checkbox" autocomplete="off"> Checkbox 3
  </label>
</div>
```

## EXAMPLE

Radio 1 (preselected)	Radio 2	Radio 3
-----------------------	---------	---------

```
<div class="btn-group" data-toggle="buttons">
  <label class="btn btn-primary active">
    <input type="radio" name="options" id="option1" autocomplete="off" checked>
      Radio 1 (preselected)
    </label>
  <label class="btn btn-primary">
    <input type="radio" name="options" id="option2" autocomplete="off"> Radio 2
  </label>
  <label class="btn btn-primary">
    <input type="radio" name="options" id="option3" autocomplete="off"> Radio 3
  </label>
</div>
```

## 方法

**\$(().button('toggle'))**

切換按下的狀態。賦予按鈕被啟用時應有的外觀樣式。

**\$(().button('reset'))**

重新設置按鈕狀態 - 這會還原按鈕文字。

**\$(().button(string))**

這會將按鈕文字更改為 `data-` 定義的文字。

```
<button type="button" id="myStateButton" data-complete-text="finished!"
  class="btn btn-primary" autocomplete="off">
  ...
</button>

<script>
  $('#myStateButton').on('click', function () {
    $(this).button('complete') // button text will be "finished!"
  })
</script>
```

## 折疊效果 collapse.js

### 關於

從折疊元素取得基礎樣式和彈性的支援，像是手風琴樣式（accordions）和巡覽樣式等。

#### 外掛相依性

折疊元件需依賴轉場外掛，請包含在你使用的 Bootstrap 版本中。

# 手風琴樣式範例

使用折疊外掛，我們建置一個簡單手風琴樣式擴充小元件：

## 譯者註

手風琴 (<http://en.wikipedia.org/wiki/Accordion>)，西方一種常用樂器，延伸意思是可折疊。讀者或者可以想成扇子。

## EXAMPLE

### 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

### Collapsible Group Item #3

```

<div class="panel-group" id="accordion" role="tablist" aria-
multiselectable="true">
  <div class="panel panel-default">
    <div class="panel-heading" role="tab" id="headingOne">
      <h4 class="panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapseOne"
aria-expanded="true" aria-controls="collapseOne">
          Collapsible Group Item #1
        </a>
      </h4>
    </div>
    <div id="collapseOne" class="panel-collapse collapse in" role="tabpanel"
aria-labelledby="headingOne">
      <div class="panel-body">
        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
creed 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>
    </div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading" role="tab" id="headingTwo">
      <h4 class="panel-title">
        <a class="collapsed" data-toggle="collapse" data-parent="#accordion"
href="#collapseTwo" aria-expanded="false" aria-controls="collapseTwo">
          Collapsible Group Item #2
        </a>
      </h4>
    </div>
    <div id="collapseTwo" class="panel-collapse collapse" role="tabpanel" aria-
labelledby="headingTwo">
      <div class="panel-body">
        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
creed 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>
    </div>
  </div>
  <div class="panel panel-default">
    <div class="panel-heading" role="tab" id="headingThree">
      <h4 class="panel-title">
        <a class="collapsed" data-toggle="collapse" data-parent="#accordion"

```

```
href="#collapseThree" aria-expanded="false" aria-controls="collapseThree">
    Collapsible Group Item #3
  </a>
</h4>
</div>
<div id="collapseThree" class="panel-collapse collapse" role="tabpanel"
aria-labelledby="headingThree">
  <div class="panel-body">
    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.
  </div>
</div>
</div>
</div>
```

它也能替換 .panel-body 為 .list-group 。

### Collapsible list group

你也可以不使用手風琴標記來使用此外掛。使用一個按鈕來切換與折疊另一個元素。

```
<button type="button" class="btn btn-danger" data-toggle="collapse" data-
target="#demo" aria-expanded="true" aria-controls="demo">
  simple collapsible
</button>

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

### 讓展開與折疊控制具有無障礙

確保有加入 aria-expanded 到控制的元素。此屬性能在螢幕閱讀器和其他輔助技術中明確地定義，現在的狀態是一個可折疊元素。如果可折疊元素預設是關閉，它應該設置 aria-expanded="false" 屬性值。如果你使用 in 類別來設置可折疊元素的預設是開啟（展開），那麼改設置為 aria-expanded="true"。基於可折疊元素是否已經被開啟或關閉，外掛會自動切換此屬性。

此外，如果你控制元素的目標是單一可折疊元素 – 即 data-target 屬性指向一個 id 選擇器 – 你可以加入額外的 aria-controls 屬性到含有 id 可折疊元素的控制元素上。主流螢幕閱讀器和其他輔助技術可以使用此屬性提供使用者便利的巡覽方式到可折疊元素。

# 使用方法

折疊外掛運用少數類別來處理繁重的工作：

- .collapse 隱藏內容
- .collapse.in 顯示內容
- 當轉場效果開始 .collapsing 會被加入，當效果結束會被移除。

這些類別可以在 `component-animations.less` 找到。

## 經由 data- 屬性

僅需的加入 `data-toggle="collapse"` 和 `data-target` 到元素上就能自動指派折疊元素的控制權。`data-target` 屬性接受一個 CSS 選擇器，以選取元素套用折疊樣式。另外，一定要加入 `collapse` 類別到可折疊元素上。如果要預設某折疊元素是開啟的，還要額外加入 `in` 類別。

若要加入手風琴樣式的群組管理到可折疊控制項，需要加入 `data-` 屬性 `data-parent="#selector"`。

### 譯者註

請參考上面範例中 `<a>` 元素的 `data-parent="#accordion"` 屬性設置，它對應的是外層的 `<div>` 的 `id="accordion"`。

## 經由 JavaScript

手動啟用可折疊效果：

```
$('.collapse').collapse()
```

## 選項

選項能透過 `data-` 屬性或 JavaScript 來傳遞。使用 `data-` 屬性，將選項名稱附加至 `data-` 後面，例如，`data-parent=""`。

名稱	類型	預設	說明
parent	selector	false	如果指定了選擇器，那麼在呈現此折疊項目時，其他元素為關閉折疊效果。（與傳統手風琴行為類似，這相依於 <code>panel</code> 類別）。
toggle	boolean	true	呼叫時切換可折疊元素的折疊效果。

## 方法

### .collapse(options)

將你的內容轉換為一個可折疊元素。接受一個選擇性參數 `object`。

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

### .collapse('toggle')

切換可折疊元素的顯示或隱藏。

### .collapse('show')

顯示可折疊元素。

### .collapse('hide')

隱藏可折疊元素。

## 事件

Bootstrap 折疊功能有公開一些事件允許監聽折疊功能。

事件類別	說明
show.bs.collapse	當 show 執行個體方法被呼叫，此事件會立即被觸發。
shown.bs.collapse	當折疊效果已經在使用者面前顯示完成（且 CSS 轉場效果已經完成），此事件會被觸發。
hide.bs.collapse	當 hide 執行個體方法被呼叫，此事件會立即被觸發。
hidden.bs.collapse	當折疊效果已經在使用者面前隱藏完成（且 CSS 轉場效果已經完成），此事件會被觸發。

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

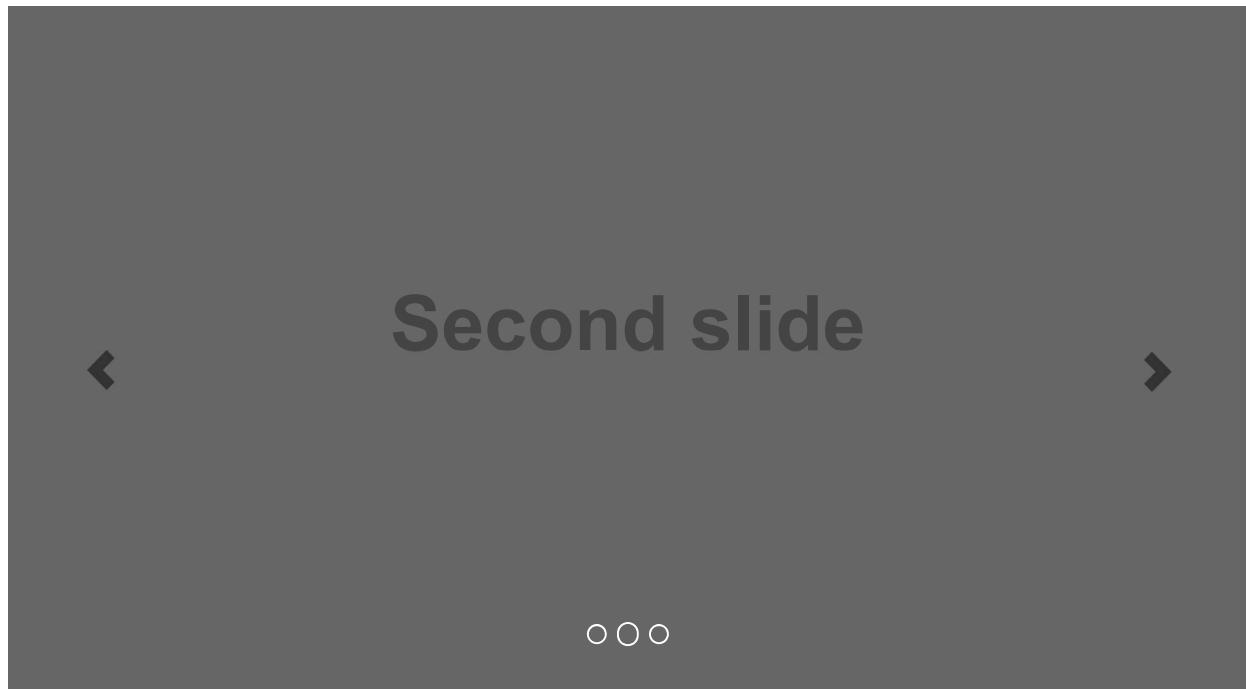
## 輪播效果 carousel.js

透過元素的循環組成幻燈片的元件，類似輪播效果。巢狀輪播不支援。

# 範例

---

EXAMPLE



```

<div id="carousel-example-generic" class="carousel slide" data-ride="carousel">
  <!-- Indicators -->
  <ol class="carousel-indicators">
    <li data-target="#carousel-example-generic" data-slide-to="0" class="active"></li>
    <li data-target="#carousel-example-generic" data-slide-to="1"></li>
    <li data-target="#carousel-example-generic" data-slide-to="2"></li>
  </ol>

  <!-- Wrapper for slides -->
  <div class="carousel-inner" role="listbox">
    <div class="item active">
      
      <div class="carousel-caption">
        ...
      </div>
    </div>
    <div class="item">
      
      <div class="carousel-caption">
        ...
      </div>
    </div>
    ...
  </div>

  <!-- Controls -->
  <a class="left carousel-control" href="#carousel-example-generic" role="button" data-slide="prev">
    <span class="glyphicon glyphicon-chevron-left" aria-hidden="true"></span>
    <span class="sr-only">Previous</span>
  </a>
  <a class="right carousel-control" href="#carousel-example-generic" role="button" data-slide="next">
    <span class="glyphicon glyphicon-chevron-right" aria-hidden="true"></span>
    <span class="sr-only">Next</span>
  </a>
</div>

```

### 無障礙問題

輪播元素一般不符合無障礙標準。如果你需要相容，請考慮其他方式來展示你的內容。

### Internet Explorer 8 & 9 不支援轉場動畫效果

Bootstrap 的動畫完全使用 CSS3，但 Internet Explorer 8 & 9 不支援所需的 CSS 屬性。因此，當使用這些版本的瀏覽器，幻燈片不會有轉場動畫效果。我們故意決定不再包含基於 jQuery 的相容性方案。

## 需要初始化 .active 元素

.active 類別需要被加入到其中一張幻燈片中。否則，輪播將不可見。

## 可選標題

在任何的 .item 內可輕鬆加入 .carousel-caption 元素來加入標題。加入的任何選擇性 HTML 將會自動對齊與格式化。

### EXAMPLE



```
<div class="item">
  
  <div class="carousel-caption">
    <h3>...</h3>
    <p>...</p>
  </div>
</div>
```

## Usage

### 多個輪播

使用 data- 屬性很容易去控制輪播的位置。data-slide 接受 prev 或 next 為關鍵字，這會改變幻燈片的位置。另外，使用 data-slide-to 可以傳遞幻燈片索引值給輪播外掛，例如，data-slide-to="2" 就可以直接播放指定索引值的幻燈片。注意，索引值由 0 開始。

## 經由 data- 屬性

使用 data- 屬性很容易去控制輪播的位置。data-slide 接受 prev 或 next 為關鍵字，這會改變幻燈片的位置。另外，使用 data-slide-to 可以傳遞幻燈片索引值給輪播外掛，例如，data-slide-to="2" 就可以直接播放指定索引值的幻燈片。注意，索引值由 0 開始。

data-ride="carousel" 屬性用來標記頁面載入後開始的輪播動畫。它不能與 JavaScript 的初始化一起組合使用在同一個輪播（多餘與不必要）。

## 經由 JavaScript

手動呼叫輪播：

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

## 選項

選項能透過 data- 屬性或 JavaScript 來傳遞。使用 data- 屬性，將選項名稱附加至 data- 後面，例如，data-interval=""。

名稱	類型	預設	說明
interval	number	5000	在自動輪播的過程中，每張幻燈片所呈現的時間。如果為 false，將不會自動輪播。
pause	string	"hover"	當滑鼠移至輪播區域時暫停輪播效果，滑鼠離開輪播區域時重新輪播效果。
wrap	boolean	true	輪播是否應該不斷循環或是停止。
keyboard	boolean	true	輪播是否要回應鍵盤事件。

## 方法

### .carousel(options)

輪播的初始化有一個選擇性選項 object 和自動開始輪播項目。

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

### .carousel('cycle')

由左向右循環播放。

### .carousel('pause')

停止循環播放。

### .carousel(number)

循環播放到指定畫面（以 0 為基礎，類似陣列）。

### .carousel('prev')

回到上一個畫面。

### .carousel('next')

移至下一個畫面。

## 事件

Bootstrap 輪播效果類別有公開一些事件允許監聽輪播功能。

以下事件額外擁有兩個屬性：

- direction : 輪播幻燈片的方向（"left" 或 "right" 其中之一）。
- relatedTarget : 正在滑動的地方作為活動項目的 DOM 元素。

事件類別	說明
slide.bs.carousel	當 slide 執行個體方法被呼叫，此事件會立即被觸發。
slid.bs.carousel	當輪播已經在使用者面前顯示完成，此事件會被觸發。

```
$('#myCarousel').on('slide.bs.carousel', function () {  
  // do something...  
})
```

## 附加巡覽 affix.js

### 範例

本文件右邊的巡覽列就是一個附加巡覽外掛的展示。

#### 譯者註

本文件最上方是主巡覽列，除首頁外，其他教學文件頁面都有右邊附加巡覽（或稱子巡覽、次巡覽）。

## 使用方法

透過 data- 屬性或你自己的 JavaScript 來手動使用附加巡覽外掛。在這兩種情境下，你必須提供

CSS 定位點和你附加巡覽內容的寬度。

## 透過 CSS 定位

附加巡覽外掛的切換在三個類別之間，每一個都代表一種特定狀態：`.affix`、`.affix-top` 和 `.affix-bottom`。因為這些類別能夠處理實際位置，你必須為這些類別提供你的樣式（獨立於此外掛）。

以下是附加巡覽如何作業：

1. 一開始，外掛會加入 `.affix-top` 以指明該元素是在最外層的位置。在這一點上，沒有 CSS 定位是必須的。
2. 滑動你想附加巡覽的元素並實際觸發附加巡覽。這裡 `.affix-top` 會被 `.affix` 替換並且設置 `position: fixed;`（這由 Bootstrap 的 CSS 提供）。
3. 如果有定義底部位移（bottom offset），滑動應該會將 `.affix` 替換為 `.affix-bottom`。由於位移是選擇性的，設置的一個要求是你也要設置相對應的 CSS。在此情況下，當有需要時可以加入 `position: absolute;`。外掛會使用 `data-` 屬性或 JavaScript 選項來決定如何定位元素的位置。

依照上述步驟，使用下面選項來設置你的 CSS。

## 經由 `data-` 屬性

很輕鬆就可以加入附加巡覽的行為到任何元素，僅需要加入 `data-spy="affix"` 到你想要偵察的元素上。使用位移（offset）來定義何時切換鎖定的元素。

```
<div data-spy="affix" data-offset-top="60" data-offset-bottom="200">
  ...
</div>
```

## 經由 JavaScript

經由 JavaScript 來呼叫附加巡覽：

```
$('#myAffix').affix({
  offset: {
    top: 100,
    bottom: function () {
      return (this.bottom = $('.footer').outerHeight(true))
    }
  }
})
```

## 選項

選項能透過 `data-` 屬性或 JavaScript 來傳遞。`data-` 屬性，將選項名稱附加至 `data-` 後面，例如，`data-offset-top="200"`。

名稱	類型	預設	說明
----	----	----	----

名稱	類型	預設	說明
offset	number   function   object	10	捲軸移動時，會從螢幕的像素去計算位移的位置。如果提供一個數字，那麼位移會直接套用到 top 和 bottom。監聽單一方 ( top 或 bottom ) 的位移，僅需提供 offset: { top: 10 } 或 offset: { top: 10, bottom: 5 } 物件。當你需要動態地提供一個位移量時，請使用函數。
target	selector   node   jQuery element	the window object	指定附加巡覽的目標元素。

## 事件

Bootstrap 附加巡覽類別有公開一些事件允許監聽附加巡覽功能。

事件類別	說明
affix.bs.affix	在該元素完成 .affix 設置前，此事件會立即被觸發。
affixed.bs.affix	在該元素完成 .affix 設置後，此事件會被觸發。
affix-top.bs.affix	在該元素完成 .affix-top 設置前，此事件會立即被觸發。
affixed-top.bs.affix	在該元素完成 .affix-top 設置後，此事件會被觸發。
affix-bottom.bs.affix	在該元素完成 .affix-bottom 設置前，此事件會立即被觸發。
affixed-bottom.bs.affix	在該元素完成 .affix-bottom 設置後，此事件會被觸發。

 Recommend 2 Share

從最好的優先排列 ▾



Start the discussion...

Be the first to comment.

 Subscribe  加入 Disqus 到你的網站 Add Disqus Add  隱私

設計與建置由有世界大愛的 @mdo (<http://twitter.com/mdo>) 和 @fat (<http://twitter.com/fat>)。  
並由 陳傳興 (/About) 翻譯。

由核心團隊 (<https://github.com/twbs?tab=members>) 與 我們的貢獻者  
(<https://github.com/twbs/bootstrap/graphs/contributors>) 的幫助下維護。

程式碼依 MIT (<https://github.com/twbs/bootstrap/blob/master/LICENSE>) 授權，英文文件依  
CC BY 3.0 (<http://creativecommons.org/licenses/by/3.0/>) 授權。  
Code licensed under MIT (<https://github.com/twbs/bootstrap/blob/master/LICENSE>),  
documentation under CC BY 3.0 (<http://creativecommons.org/licenses/by/3.0/>).

翻譯版本 v3.3.1 · GitHub (<https://github.com/twbs/bootstrap>) · 範例  
(</bs3/Gettingstarted#examples>) · v2.3.2 (</bs2>) · 關於 (</bs3/About>) · 官方精選  
(<http://expo.getbootstrap.com>) · 官方Blog (<http://blog.getbootstrap.com>) · 問題  
(<https://github.com/twbs/bootstrap/issues>) · 發行版本  
(<https://github.com/twbs/bootstrap/releases>)