# Bootstrap插件

**目 录**

[Bootstrap插件 1](#_Toc14382)

[1 模态弹出框（Modals） 5](#_Toc6442)

[1.1 模态弹出框--结构分析 5](#_Toc24092)

[1.2 模态弹出框--实现原理解析 6](#_Toc17159)

modal

modal-dialog

modal-content

modal-header

modal-body

modal-footer

[1.3 模态弹出框--触发模态弹出窗2种方法 9](#_Toc30379)

button[type="button"].btn.btn-primary[data-tggle="modal"][data-target="#模态窗口的id"]

**表示关闭**

modal-header>button[type="button"].colse[data-dismiss="modal"]{&times;}

[1.4 模态弹出框--为弹出框增加过度动画效果 11](#_Toc31616)

modal.fade

[1.5 模态弹出框--模态弹出窗的使用（data-参数说明） 12](#_Toc30958)

moda[data-backdrop=""]

true 点击灰色背景会关闭摸态框

flase 使灰色背景不存在

static 点击灰色背景不关闭摸态框

moda[data-keyboard=""]

true 在input框里面，点击ecs键会关闭模态窗

flase 与true相反

[1.6 模态弹出框--模态弹出窗的使用（JavaScript触发） 13](#_Toc23374)

[1.7 模态弹出框--JavaScript触发时的参数设置（一） 14](#_Toc26851)

[1.8 模态弹出框--JavaScript触发时的参数设置（二） 15](#_Toc5114)

[2下拉菜单（Dropdown） 17](#_Toc357)

[2.1 下拉菜单--属性声明式方法（一） 17](#_Toc28286)

[2.2 下拉菜单--属性声明式方法（二) 18](#_Toc17106)

[2.3下拉菜单--JavaScript触发方法 20](#_Toc32748)

[3 滚动监控器 21](#_Toc13200)

[3.1滚动监控器--滚动监控器的设计 22](#_Toc19516)

[3.2 滚动监控器--声明属性触发滚动监控 25](#_Toc18974)

[3.3 滚动监控器--JavaScript方法触发滚动监控器 26](#_Toc4281)

[4 选项卡（Tabs） 27](#_Toc29782)

[4.1 选项卡--选项卡的结构 28](#_Toc31394)

[4.2 选项卡--触发切换效果 29](#_Toc3589)

[4.4 选项卡--胶囊式选项卡（nav-pills） 32](#_Toc29686)

[4.5 选项卡--JavaScript触发方法 33](#_Toc26889)

[5 提示框(Tooltip) 34](#_Toc16264)

[5.1 提示框--结构 34](#_Toc23443)

[5.2 提示框--JS触发提示框方法 36](#_Toc9932)

[5.3 提示框--JS设置参数方法 37](#_Toc11583)

[6 弹出框（Popover） 39](#_Toc20226)

[6.1 弹出框--弹出框的结构 39](#_Toc16921)

[6.2 弹出框--触发弹出框的方法 41](#_Toc14494)

[6.3弹出框--自定义结构属性 43](#_Toc6877)

[7 警告框--结构与样式 45](#_Toc17193)

[7.1 警告框--使用声明式触发警告框 45](#_Toc15324)

[7.3 警告框--JavaScript触发警告框 46](#_Toc8428)

[8 按钮 47](#_Toc22644)

[8.1 按钮插件--按钮加载状态 48](#_Toc16671)

[9 手风琴（Collapse） 48](#_Toc20234)

[9.1 手风琴--手风琴结构 49](#_Toc11384)

[9.2手风琴--声明式触发手风琴（一） 51](#_Toc8657)

[10 图片轮播（Carousel） 57](#_Toc28493)

[10 .1 图片轮播--轮播图片的设计 57](#_Toc15090)

[10.2图片轮播--声明式触轮播图的播放 63](#_Toc17831)

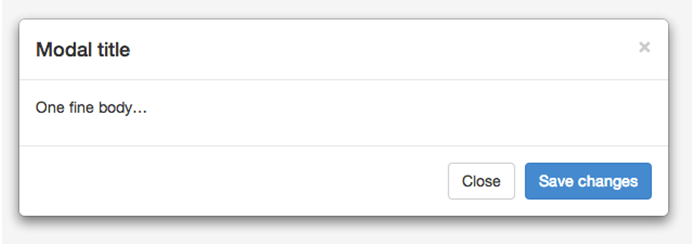
[10.4 图片轮播--JavaScript触发方法 66](#_Toc32463)

[11 . 固定定位（Affix） 68](#_Toc8431)

[11. 1固定定位--声明式触发固定定位 68](#_Toc1395)

# 1 模态弹出框（Modals）

在 Bootstrap 框架中把模态弹出框统一称为 Modal 。这种弹出框效果在大多数 Web 网站的交互中都可见。比如点击一个按钮弹出一个框，弹出的框可能是一段文件描述，也可能带有按钮操作，也有可能弹出的是一张图片。如下图所示：



## 1.1 模态弹出框--结构分析

Bootstrap框架中的模态弹出框，分别运用了“**modal**”、“**modal-dialog**”和“**modal-content**”样式，**而弹出窗真正的内容都放置在“modal-content”中**，其主要又包括三个部分：

**☑ 弹出框头部，一般使用“modal-header”表示，主要包括标题和关闭按钮**

**☑ 弹出框主体，一般使用“modal-body”表示，弹出框的主要内容**

**☑ 弹出框脚部，一般使用“modal-footer”表示，主要放置操作按钮**

模态弹出窗的结构如下：

<div class="modal show">

<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">模态弹出窗标题</h4>

</div>

<div class="modal-body">

<p>模态弹出窗主体内容</p>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-default" data-dismiss="modal">关闭</button>

<button type="button" class="btn btn-primary">保存</button>

</div>

</div><!-- /.modal-content -->

</div><!-- /.modal-dialog -->

</div><!-- /.modal -->

弹出窗的主体样式实现：

但是对于一个模态弹出窗而言，modal-content才是样式的关键。其主要设置了弹出窗的边框、边距、背景色和阴影等样式。

## 1.2 模态弹出框--实现原理解析

**实现原理解析：**

bootstrap中的“模态弹出框”有以下几个特点：

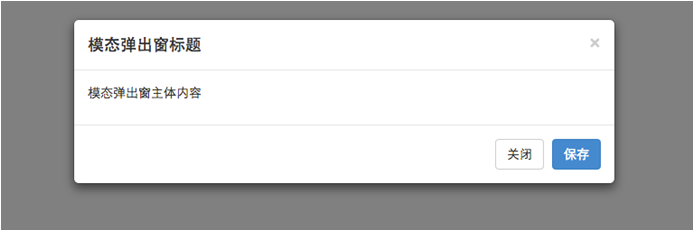
**1、**模态弹出窗是固定在浏览器中的。

**2、**单击右侧全屏按钮，在全屏状态下，模态弹出窗宽度是自适应的，而且modal-dialog水平居中。

**3、**当浏览器视窗大于768px时，模态弹出窗的宽度为600px。

**蒙板样式实现：**

大家或许注意到了，在做模态弹出窗时，底部常常会有一个**透明的黑色蒙层**效果，如下图所示：

[](http://img.mukewang.com/541a428200016e9806930230.jpg)

在Bootstrap框架中为模态弹出窗也添加了一个这样的蒙层样式“**modal-backdrop**”，只不过他默认情况下是全屏黑色的。

****两种尺寸选择：****

除此之外，Bootstrap框架还为模态弹出窗提供了不同尺寸，一个是**大尺寸样式“modal-lg”，另一个是小尺寸样式“modal-sm”**。其结构上稍做调整：

<!-- 大尺寸模态弹出窗 -->

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

<divclass="modal-dialog ****modal-lg****">

<divclass="modal-content"> ... </div>

</div>

</div>

<!-- 小尺寸模态弹出窗 -->

<divclass="modal fade bs-example-modal-sm"tabindex="-1"role="dialog" aria-labelledby="mySmallModalLabel" aria-hidden="true">

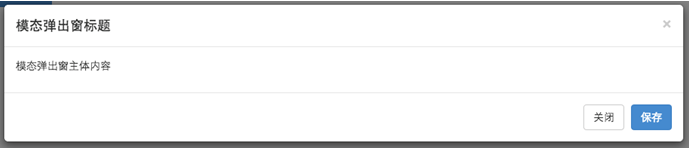
<divclass="modal-dialog ****modal-sm****">

<divclass="modal-content"> ... </div>

</div>

</div>

来简单的看一个示例效果：

[](http://img.mukewang.com/541a42f4000194dc06890148.jpg)

[](http://img.mukewang.com/541a430800019bf006220370.jpg)

## 1.3 模态弹出框--触发模态弹出窗2种方法

众所周知，模态弹出窗在页面加载完成时，是被隐藏在页面中的，只有通过一定的动作（事件）才能触发模态弹出窗的显示。在Bootstrap框架中实现方法有2种，接下来分别来介绍这2种触发模态弹出窗的使用方法。

**声明式触发方法：**

**方法一：**模态弹出窗声明，只需要自定义两个必要的属性：**data-toggle**和**data-target**（bootstrap中声明式触发方法一般依赖于这些自定义的data-xxx 属性。比如data-toggle="" 或者 data-dismiss=""）。例如：

<!-- 触发模态弹出窗的元素 -->

<button type="button" data-toggle="modal" data-target="#mymodal" class="btn btn-primary">点击我会弹出模态弹出窗</button>

<!-- 模态弹出窗 -->

<div class="modal fade" id="mymodal">

<div class="modal-dialog">

<div class="modal-content">

<!-- 模态弹出窗内容 -->

</div>

</div>

</div>

**注意以下事项：**

1、data-toggle必须设置为modal(toggle中文翻译过来就是触发器)；

2、data-target可以设置为CSS的选择符，也可以设置为模态弹出窗的ID值，一般情况设置为模态弹出窗的ID值，因为ID值是唯一的值。

**方法二：**触发模态弹出窗也可以是一个**链接<a>元素**，那么可以使用链接元素自带的**href属性替代data-target属性**，如：

<!-- 触发模态弹出窗的元素 -->

<a data-toggle="modal" href="#mymodal" class=" btn btn-primary" >点击我会弹出模态弹出窗</a>

<!-- 模态弹出窗 -->

<div class="modal fade" id="mymodal" >

<div class="modal-dialog" >

<div class="modal-content" >

<!-- 模态弹出窗内容 -->

</div>

</div>

</div>

**不过建议还是使用统一使用data-target的方式来触发。**

点击按钮就能触发弹出窗：

[](http://img.mukewang.com/541a76680001447f06640188.jpg)

## 1.4 模态弹出框--为弹出框增加过度动画效果

为模态弹出框增加**过度动画效果**：

可通过**给“.modal”增加类名“fade”**为模态弹出框增加一个过渡动画效果。

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

小的模态弹出窗

</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 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">模态弹出窗标题</h4>

</div>

<div class="modal-body">

<p>模态弹出窗主体内容</p>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-default" data-dismiss="modal">关闭</button>

<button type="button" class="btn btn-primary">保存</button>

</div>

</div>

</div>

</div>

## 1.5 模态弹出框--模态弹出窗的使用（data-参数说明）

除了通过data-toggle和data-target来控制模态弹出窗之外，Bootstrap框架针对模态弹出框还提供了其他**自定义data-属性**，来控制模态弹出窗。比如说:是否有灰色背景modal-backdrop，是否可以按ESC键关闭模态弹出窗。有关于Modal弹出窗自定义属性相关说明如下所示：

[](http://img.mukewang.com/544f09480001d6c409000872.jpg)

## 1.6 模态弹出框--模态弹出窗的使用（JavaScript触发）

**JavaScript触发方法**

除了使用自定义属性触发模态弹出框之外，还可以通过JavaScript方法来触发模态弹出窗。通过给一个元素一个事件，来触发。比如说给一个**按钮一个单击事件**，然后触发模态弹出窗。如下面的一个简单示例：

<!-- 触发模态弹出窗元素 -->

<button class="****btn**** btn-primary" type="button">点击我</button>

<!-- 模态弹出窗内容 -->

<div class="modal" ****id="mymodal"****>

<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">模态弹出窗标题</h4>

</div>

<div class="modal-body">

<p>模态弹出窗主体内容</p>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-default" data-dismiss="modal">关闭</button>

<button type="button" class="btn btn-primary">保存</button>

</div>

</div><!-- /.modal-content -->

</div><!-- /.modal-dialog -->

</div><!-- /.modal -->

JavaScript触发的弹出窗代码：

$(function(){

$("****.btn****").click(function(){

$("****#mymodal****").modal();

});

});

运行效果如下：

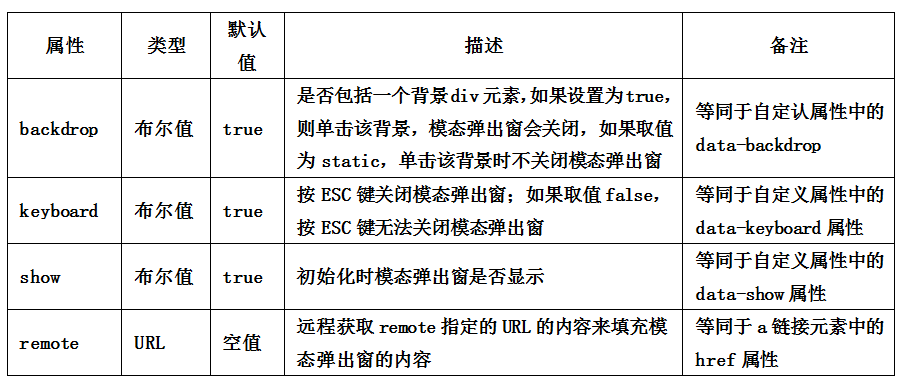
[](http://img.mukewang.com/541a7d580001e8a606490343.jpg)

## 1.7 模态弹出框--JavaScript触发时的参数设置（一）

使用JavaScript触发模态弹出窗时，Bootstrap框架提供了一些设置，主要包括**属性设置**、**参数设置**和**事件设置**。

属性设置

模态弹出窗默认支持的自定义属性主要有：

[](http://img.mukewang.com/544f0bd50001b34409020384.jpg)

比如你不想让用户按ESC键关闭模态弹出窗，你就可以这样做：

$(function(){

$(".btn").click(function(){

$("#mymodal").modal({

keyboard:false

});

});

});

## 1.8 模态弹出框--JavaScript触发时的参数设置（二）

我们接着对**参数设置**和**事件设置**进行介绍。

参数设置：

在Bootstrap框架中还为模态弹出窗提供了三种参数设置，具体说明如下：

|  |  |  |
| --- | --- | --- |
| 参数 | 使用方法 | 描述 |
| toggle | $(“#mymodal”).modal(“toggle”) | 触发时，反转模态弹出窗的状态。如果模态弹出窗是显示的，则关闭；反之，如果模态弹出窗是关闭的，则显示 |
| show | $(“#mymodal”).modal(“show”) | 触发时，显示模态弹出窗 |
| hide | $(“#mymodal”).modal(“hide”) | 触发时，关闭模态弹出窗 |

**事件设置：**

模态弹出窗还支持四种类型的事件，分别是模态弹出窗的弹出前、弹出后，关闭前、关闭后，具体描述如下：

|  |  |
| --- | --- |
| 事件类型 | 描述 |
| **show.bs.modal** | 在show方法调用时立即触发（尚未显示之前）；如果单击了一个元素，那么该元素将作为事件的relatedTarget属性 |
| shown.bs.modal | 该事件在模态弹出窗完全显示给用户之后（并且等CSS动画完成之后）触发；如果单击了一个元素，那么该元素将作为事件的relatedTarget事件 |
| hide.bs.modal | 在hide方法调用时（但还未关闭隐藏）立即触发 |
| hidden.bs.modal | 该事件在模态弹出窗完全隐藏之后（并且CSS动画漂完成之后）触发 |

调用方法也非常简单：

$('#myModal').on('hidden.bs.modal', function (e) {

// 处理代码...

})

# **2下拉菜单（Dropdown）**

在介绍Bootstrap的组件的时候，只介绍了下拉菜单的结构和表现形式（样式风格），在这一节中主要介绍怎么结合JavaScript代码实现交互效果。

和模态弹出窗一样，触发下拉菜单方式有两种，一种是**属性声明式用法**，另一种是**JavaScript方法**。下面几个小节分别来看这两种方法是如何触发下拉菜单。

## 2.1 下拉菜单--属性声明式方法（一）

**属性声明式方法：**

一般下拉菜单都是出现在导航条中，比如下图的一个效果，用户点击带有三角形的菜单项都会弹出下拉菜单项：

[](http://img.mukewang.com/541a81cc0001caab06720153.jpg)

<div class="navbar navbar-default" id="navmenu">

<a href="##" class="navbar-brand">W3cplus</a>

<ul class="nav navbar-nav">

<li class="dropdown">

<a href="##" data-toggle="dropdown" class="dropdown-toggle" role="button" id="tutorial">教程<b class="caret"></b></a>

<ul class="dropdown-menu" role="menu" aria-labelledby="tutorial">

<li role="presentation"><a href="##">CSS3</a></li>

<li role="presentation"><a href="##">HTML5</a></li>

<li role="presentation"><a href="##">Sass</a></li>

</ul>

</li>

<li><a href="##">前端论坛</a></li>

<li><a href="##">关于我们</a></li>

</ul>

</div>

## 2.2 下拉菜单--属性声明式方法（二)

从上一个示例，我们可以知道，用户只需要点击有向下三角形的按钮链接或者直接点击三角形就会弹出下拉菜单。实现这个效果，都是依赖于HTML相关元素自定义的属性完成。所以在编写HTML结构的时候必须满足下面的规则：

  ☑ 被点击的菜单项链接或按钮需要添加自定义属性**data-toggle="dropdown"**

<div class="navbar navbar-default" id="navmenu">

<a href="##" class="navbar-brand">W3cplus</a>

<ul class="nav navbar-nav">

<li class="dropdown">

<a href="##" data-toggle="dropdown" class="dropdown-toggle" role="button" id="tutorial">教程<b class="caret"></b></a>

<ul class="dropdown-menu" role="menu" aria-labelledby="tutorial">

<li role="presentation"><a href="##">CSS3</a></li>

...

</ul>

</li>

<li><a href="##">前端论坛</a></li>

<li><a href="##">关于我们</a></li>

</ul>

</div>

****实现下拉菜单原理：****

Dropdown插件加载时，对所有带 有“data-toggle=dropdown”样式的元素绑定了事件，用户单击带有“data-toggle=dropdown”样式的链接或按钮时， 会触发JavaScript事件代码。当用户点击带有“data-toggle=dropdown”样式的链接或按钮时，下拉菜单的父容器（上面的示例是 “<li class="dropdown">”）会添加一个open类名，此时下拉菜单显示；再次单击时，JavaScript会删除刚添加的open类 名，此时下拉菜单将隐藏。

**简单的说，要制作下拉菜单，其结构基本如下：**

<div class="dropdown">

<a data-toggle="dropdown" href="#">下拉菜单触发器</a>

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

</div>

如果触发下拉菜单的元素是一个链接元素，为了**避免点击链接，页面跳到顶部**，可以使用**data-target="#"来替代href="#"**：

<div class="dropdown">

<a id="dLabel" role="button" data-toggle="dropdown" data-target="#" href="/page.html"> Dropdown <span class="caret"></span></a>

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

</div>

## 2.3下拉菜单--JavaScript触发方法

和模态弹出窗一样，Bootstrap框架中的下拉菜单也支持JavaScript方法触发下拉菜单显示。使用JavaScript触发下拉菜单和声明式原理是基本类似的，不同的是使用JavaScript方法可以按照自己的方式省略一些元素或者样式。同样用一个简单的示例来做演示：

<ul class="nav nav-pills">

<li class="dropdown">

<a href="##" class="dropdown-toggle" role="button" id="tutorial">教程<b class="caret"></b></a>

<ul class="dropdown-menu" role="menu" aria-labelledby="tutorial">

<li role="presentation"><a href="##">CSS3</a></li>

<li role="presentation"><a href="##">HTML5</a></li>

<li role="presentation"><a href="##">Sass</a></li>

</ul>

</li>

<li class="active"><a href="##">前端论坛</a></li>

<li><a href="##">关于我们</a></li>

</ul>

使用JavaScript调用dropdown()方法后，单击激活按钮，会弹出下拉菜单，再次单击的时候会收起下拉菜单。

$(function(){

$(".dropdown-toggle").dropdown();

})

还可以使用参数“toggle”。当下拉菜单隐藏时，调用**dropdown(“toggle”)方法**可以显示下拉菜单，反之，如果下拉菜单显示时，调用dropdown(“toggle”)方法可以让下拉菜单隐藏。

$(function(){

$(".dropdown-toggle").dropdown("toggle");

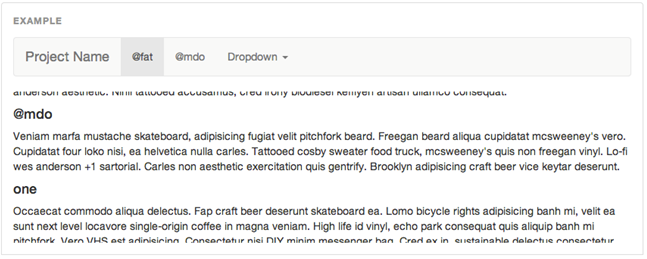
})

不过使用该参数，每次单击都要两次toggle，就会一直是一个不变的状态。所以，一般情况下，使用示例中不带参数的方法。

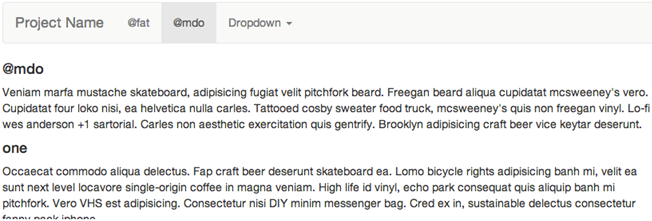
# 3 滚动监控器

滚动监控器是Bootstrap提供的非常实用的JavaScript插件，被广泛应用到Web开发中。其表现形式是：

**1、当用户鼠标滚动时，滚动条的位置会自动更新导航条中相应的导航项**。如Bootstrap官网提供的示例：

[](http://img.mukewang.com/541a88ea0001ee0006470259.jpg)

**2、用户拖动滚动条，当滚动到@mdo时，上面的@mdo导航项就会高亮显示：**

[](http://img.mukewang.com/541a89290001304506520221.jpg)

这是因为该插件可以自动检测滚动条到达哪个位置，然后在需要高亮的菜单元素上加了一个“active”样式。

# 3.1滚动监控器--滚动监控器的设计

在Web页面中实现Bootstrap滚动监控器其实非常简单，接下来我们一步一步来看滚动监控器是如何设计的。  
 **第一步：**使用滚动监控器，首要的条件是在页面中加载对应的插件。在这里你可以加载合并好的bootstrap.js或者其独立的插件文件scrollspy.js。这里以加载合并好的js为例：

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>

<script src="//maxcdn.bootstrapcdn.com/bootstrap/3.2.0/js/bootstrap.min.js"></script>

**第二步：**设计一个带有下拉菜单的导航条。分别为导航条和下拉菜单项定义一个锚点链接，锚点分别为“#blog”,“html”,“#css”,“#sass”,“#js”,“php”,“#about”。同时为导航条定义一个id值“navbar-menu”（id名称可自由定义），方便滚动监控。

<nav role="navigation" class="navbar navbar-default navbar-static" id="navbar-example2">

<div class="container-fluid">

<div class="navbar-header">

<button data-target=".bs-example-js-navbar-scrollspy" data-toggle="collapse" type="button" class="navbar-toggle collapsed">

<span class="sr-only">Toggle navigation</span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

<a href="#" class="navbar-brand">Project Name</a>

</div>

<div class="collapse navbar-collapse bs-example-js-navbar-scrollspy">

<ul class="nav navbar-nav">

<li><a href="#fat">@fat</a></li>

<li><a href="#mdo">@mdo</a></li>

<li class="dropdown">

<a aria-expanded="false" role="button" data-toggle="dropdown" class="dropdown-toggle" id="navbarDrop1" href="#">Dropdown <span class="caret"></span></a>

<ul aria-labelledby="navbarDrop1" role="menu" class="dropdown-menu">

<li><a tabindex="-1" href="#one">one</a></li>

<li><a tabindex="-1" href="#two">two</a></li>

<li class="divider"></li>

<li><a tabindex="-1" href="#three">three</a></li>

</ul>

</li>

</ul>

</div>

</div>

</nav>

**第三步：**设计监控对象。这里将监控对象内容都放置在一个div名为scrollspy（**这个类名可自由定义**）的容器中，其中放了多个子内容框。每个子内容框有一个标题，而且**每个标题的ID值与导航菜单项中的锚点链接名相对应**，并且注意加入“**data-target="#navbar-menu**"”属性（这个属性值要与前面的nav标签的id名称保持一致）

<div class="scrollspy" data-target="#navbar-menu">

<h4 id="blog">Blog</h4>

<p>…</p>

<h4 id="html">Html</h4>

<p>…</p>

<h4 id="css">CSS</h4>

<p>…</p>

<h4 id="sass">Sass</h4>

<p>…</p>

<h4 id="js">JavaScript</h4>

<p>…</p>

<p>…</p>

<h4 id="php">PHP</h4>

<p>…</p>

<p>…</p>

<h4 id="about">About</h4>

<p>…</p>

<p>…</p>

</div>

第四步：为监控对象定义样式，设置容器scrollspy大小（设置高度目的是为了产生垂直滚动条）。

.scrollspy{

height:500px;

font-size:20px;

overflow:auto;

}

## 3.2 滚动监控器--声明属性触发滚动监控

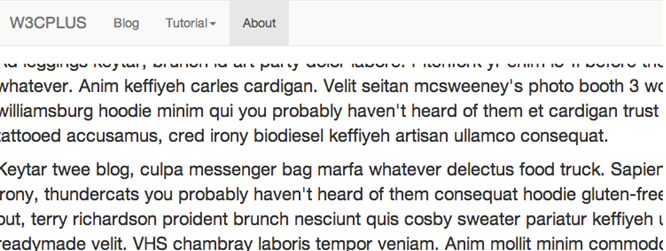
为监控对象设置被监控的data属性：**data-spy="scroll"**，指定监控的导航条：**data-target="#navbar-menu"。**同时定义监控过程中滚动条偏移位置**data-offset="60"**。代码如下：

<div class="scrollspy" data-spy="scroll" data-target="****#navbar-menu****" data-offset="60">

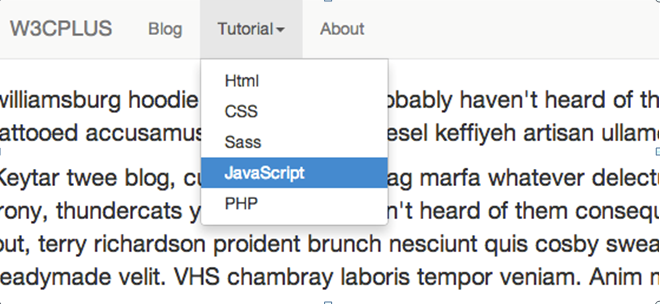
…

</div>

现在浏览器中预览，则可以看到当滚动div.scrollspy的滚动条时，导航条会实时监控并更新当前被激活的菜单项，如下图所示：

[](http://img.mukewang.com/541a8b520001ac3006660251.jpg)

如果导航里有下拉菜单，并且滚动区域的内容到达下拉菜单子项所对应的区域，除了菜单高亮以外，子菜单的父元素dropdown也会高亮，如下图所示：

[](http://img.mukewang.com/541a8b6b0001b46406600304.jpg)

## 3.3 滚动监控器--JavaScript方法触发滚动监控器

在Bootstrap框架中，使用JavaScript方法触发滚动监控器相对来说较为简单，只需要指定两个容器的名称即可。比如下面的结构：

<nav id="navbar-menu" class="navbar navbar-default navbar-static" role="navigation">

…

</nav>

<div class="scrollspy" id="scrollspy">

…

</div>

JavaScript触发可以这样写：

$(function(){

$("#scrollspy").scrollspy({

target: "#navbar-menu"

});

})

Bootstrap的滚动监控还提供了一个方法scrollspy("refresh")。当滚动监控所作用的DOM有增加或删除页面元素的操作时，需要调用下面的refresh方法：

$(function(){

$("[data-spy='scroll']").each(function(){

var $spy=$(this).scrollspy("refresh");

})

})

需要注意的是，这种refresh方法只对声明式用法有效。另外滚动监控除了options参数“target”之外，还提供了一个offset参数，此参数默认值为10。默认情况下，滚动内容距离滚动容器10px以内的话，就高一片面前显示所对应的菜单项。

# 4 选项卡（Tabs）

选项卡Tabs是Web中一种非常常用的功能。用户点击或悬浮对应的菜单项，能切换出对应的内容。如下图所示：

[](http://img.mukewang.com/5491296400019ce104700166.jpg)

默认显示“公告”对应的内容，当用户鼠标悬浮在“规则”菜单项时，显示对应规则的内容：

[](http://img.mukewang.com/5491298e0001c7ef04680166.jpg)

依此类推，通过这个效果可以得知，Bootstrap框架中的选项卡主要有两部分内容组成：

**选项卡组件（也就是菜单组件），对应的是 Bootstrap的 nav-tabs）**

**底部可以切换的选项卡面板，在 Bootstrap 中通常 tab-pane 来表示**

## 4.1 选项卡--选项卡的结构

一个选项卡主要包括两个部分，其一是菜单项，其二是内容面板。拿下面的示例来做演示。其HTML结构如下：

<!-- 选项卡组件（菜单项nav-tabs）-->

<ul id="myTab" class="nav nav-tabs" role="tablist">

<li class="active"><a href="#bulletin" role="tab">公告</a></li>

<li><a href="#rule" role="tab">规则</a></li>

<li><a href="#forum" role="tab">论坛</a></li>

<li><a href="#security" role="tab">安全</a></li>

<li><a href="#welfare" role="tab">公益</a></li>

</ul>

<!-- 选项卡面板 -->

<div id="myTabContent" class="**tab-content**">

<div class="tab-pane fade in active" id="bulletin">公告内容面板</div>

<div class="tab-pane fade" id="rule">规则内容面板</div>

<div class="tab-pane fade" id="forum">论坛内容面板</div>

<div class="tab-pane fade" id="security">安全内容面板</div>

<div class="tab-pane fade" id="welfare">公益内容面板</div>

</div>

**关键一点，选项卡中链接的锚点要与对应的面板内容容器的ID相匹配。**

在Bootstrap框架中选项卡nav-tabs已带有样式，前面在介绍导航一节中有详细介绍。而对于面板内容tab-pane都是隐藏的，只有当前面板内容才是显示的。

**但这个Tabs没有任何交互功能，点击选项卡不会进行内容的切换。**

## 4.2 选项卡--触发切换效果

同样的，选项卡也定义data属性来触发切换效果。当然前提你也要先加载bootstrap.js或者是tab.js。声明式触发选项卡需要满足以下几点要求：  
  1、**选项卡导航链接中要设置 data-toggle="tab"**

  2、**并且设置 data-target="对应内容面板的选择符(一般是ID)";**

**href 属性可以代替data-target自定义属性，data-target优先级高于href，只有在data-target为空时，href才有用。**

     如果是链接的话，还可以通过 href="对应内容面板的选择符(一般是ID)"

     主要起的作用是用户点击的时候能找到该选择符所对应的面板内容 tab-pane。

  3、**面板内容统一放在 tab-content 容器中，而且每个内容面板 tab-pane 都需要设置一个独立的选择符（最好是ID）与选项卡中的 data-target 或 href 的值匹配**。

代码如下所示：

<!-- 选项卡组件（菜单项nav-tabs）-->

<ul id="myTab" class="nav nav-tabs" role="tablist">

<li class="active"><a href="#bulletin" role="tab" data-toggle="tab">公告</a></li>

<li><a href="#rule" role="tab" data-toggle="tab">规则</a></li>

<li><a href="#forum" role="tab" data-toggle="tab">论坛</a></li>

<li><a href="#security" role="tab" data-toggle="tab">安全</a></li>

<li><a href="#welfare" role="tab" data-toggle="tab">公益</a></li>

</ul>

<!-- 选项卡面板 -->

<div id="myTabContent" class="tab-content">

<div class="tab-pane fade in active" id="bulletin">公告内容面板</div>

<div class="tab-pane fade" id="rule">规则内容面板</div>

<div class="tab-pane fade" id="forum">论坛内容面板</div>

<div class="tab-pane fade" id="security">安全内容面板</div>

<div class="tab-pane fade" id="welfare">公益内容面板</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541a943500019cf806540176.jpg)4.3 选项卡--为选择卡添加fade样式

为了让面板的隐藏与显示在切换的过程效果更流畅，可以在面板中添加类名 **fade**，让其产生渐入的效果。

在添加 fade 样式时，**最初的默认显示的内容面板一定要记得加上 in 类名**，不然其内容用户无法看到。如下面的示例所示。

<!-- 选项卡组件（菜单项nav-tabs）-->

<ul id="myTab" class="nav nav-tabs" role="tablist">

<li class="active"><a href="#bulletin" role="tab" data-toggle="tab">公告</a></li>

<li><a href="#rule" role="tab" data-toggle="tab">规则</a></li>

<li><a href="#forum" role="tab" data-toggle="tab">论坛</a></li>

<li><a href="#security" role="tab" data-toggle="tab">安全</a></li>

<li><a href="#welfare" role="tab" data-toggle="tab">公益</a></li>

</ul>

<!-- 选项卡面板 -->

<div id="myTabContent" class="tab-content">

<div class="tab-pane fade in active" id="bulletin">公告内容面板</div>

<div class="tab-pane fade" id="rule">规则内容面板</div>

<div class="tab-pane fade" id="forum">论坛内容面板</div>

<div class="tab-pane fade" id="security">安全内容面板</div>

<div class="tab-pane fade" id="welfare">公益内容面板</div>

</div>

## 4.4 选项卡--胶囊式选项卡（nav-pills）

在Bootstrap除了可以让 nav-tabs 具有选项卡的切换功能之外，还可以对胶囊式 nav-pills 导航也具有选项卡的功能。我们**只需要将 nav-tabs 换成 nav-pills。ss**

下面的示例是将上面的 nav-tabs 换成 nav-pills：

<!-- 选项卡组件（菜单项nav-pills）-->

<ul id="myTab" class="nav nav-pills" role="tablist">

<li class="active"><a href="#bulletin" role="tab" data-toggle="tab">公告</a></li>

<li><a href="#rule" role="tab" data-toggle="tab">规则</a></li>

<li><a href="#forum" role="tab" data-toggle="tab">论坛</a></li>

<li><a href="#security" role="tab" data-toggle="tab">安全</a></li>

<li><a href="#welfare" role="tab" data-toggle="tab">公益</a></li>

</ul>

<!-- 选项卡面板 -->

<div id="myTabContent" class="tab-content">

<div class="tab-pane fade in active" id="bulletin">公告内容面板</div>

<div class="tab-pane fade" id="rule">规则内容面板</div>

<div class="tab-pane fade" id="forum">论坛内容面板</div>

<div class="tab-pane fade" id="security">安全内容面板</div>

<div class="tab-pane fade" id="welfare">公益内容面板</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541a94910001649c06150198.jpg)

## 4.5 选项卡--JavaScript触发方法

除了在HTML设置 data-toggle 来触发选项卡之外，还可以通过JavaScript直接调用。

**调用方法:**

在每个链接的单击事件中调用**tab("show")**方法，显示对应的标签面板内容。针对上面的示例，**删除HTML中自定义的 data-toggle="tab"的属性**，然后通过下面的脚本来调用：

$(function(){

$("#myTab a").click(function(){

$(this).tab("show");

});

})

# 5 提示框(Tooltip)

提示框(tooltiip)效果也是Web常见的一种效果，如下图所示：

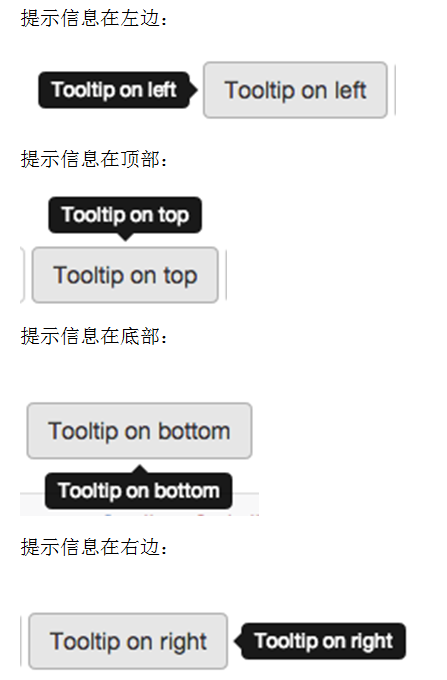
[](http://img.mukewang.com/54912aae00016b9806910295.jpg)

当用户鼠标悬浮在“当天13:00”上面时，就会出现浅黄色的提示信息框，常常把这种效果称之为提示框。

## 5.1 提示框--结构

Bootstrap框架中的tooltip的插件提供了**四种不同的风格**：

**提示信息在左边：**

[](http://img.mukewang.com/54912dd50001277104470680.jpg)

在Bootstrap框架中的提示框，结构非常简单，常常使用的是按钮<button>标签或者链接<a>标签来制作。不管是使用按钮还是链接来制作提示框，他们都有一个共性：

通过**title** 属性的值来定义提示信息(也可以使用自定义属性**data-original-title**来设置提示信息)。

通过 **data-placement** 自定义属性来控制提示信息框的位置，根据四种不同的位置，data-placement具有四个值：top、right、bottom和left，分别表示提示框出现的位置在顶部、右边、底部和左边。

还有一个最重要的参数不可缺少，**data-toggle="tooltip"。**

<button type="button"

class="btnbtn-default"

data-toggle="tooltip"

data-placement="left"

data-original-title="提示框居左">

提示框居左

</button>

<button type="button"

class="btnbtn-default"

data-toggle="tooltip"

data-placement="top"

data-original-title="提示框在顶部">

提示框在顶部

</button>

<button type="button"

class="btnbtn-default"

data-toggle="tooltip"

data-placement="bottom"

data-original-title="提示框在底部">

提示框在底部

</button>

<button type="button"

class="btnbtn-default"

data-toggle="tooltip"

data-placement="right"

data-original-title="提示框居右">

提示框居右

</button>

**需要特别注意的是：**

1、如果同时设置了**data-original-title 和 title** 定义提示信息，那么 **data-****original-title 的优先级要高于 title**。只有 data-original-title 值为空时，才会取 title 的值做为提示信息的内容。

2、Bootstrap框架中的提示框的触发方式和前面介绍的插件略有不同**。不能直接通过自定义的属性 data- 来触发**。必须得依赖于JavaScript的代码触发。

## 5.2 提示框--JS触发提示框方法

Bootstrap框架中的提示框的触发方式和前面介绍的插件略有不同。不能直接通过自定义的属性 data- 来触发。必须得依赖于JavaScript的代码触发。

**最简单的触发方式如下：**

$(function(){

$('[data-toggle="tooltip"]').tooltip();

});

除此之外，也可以单独指定一个元素，在该元素上调用提示框，并且可能通过JavaScript的各种参数来实现，无需定义一些 data 属性，如：

$(function(){

$('#myTooltip').tooltip({

title:"我是一个提示框，我在顶部出现",

placement:'top'

  });

});

**运行的效果如下：**

[](http://img.mukewang.com/541a9cc30001a1be04650139.jpg)

## 5.3 提示框--JS设置参数方法

除了在 html 代码中使用 data- 设置提示框参数，还可以使用 JavaScript 来设置提示框参数，主要包括：

[](http://img.mukewang.com/549141cf000114f911420830.jpg)

表格中的任何属性，在使用 JavaScript 触发提示框时，都可以被调用。

例子：

<script>

$(function(){

$('#myTooltip1').tooltip({

title:"我是一个提示框，我在顶部出现",

placement:"top"

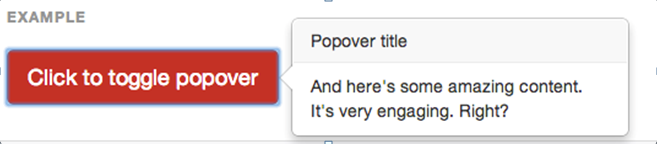
});

});

</script>

# **6 弹出框（Popover）**

弹出框(Popover)仅从外表上看，和前面介绍的提示框（Tooltip）长得差不多，如下所示：

[](http://img.mukewang.com/541a9e5e0001929606570144.jpg)

**不同的是：弹出框除了有标题 title 以外还增加了内容 content 部分**。这个在提示框中是没有的。而对于两者有何区别呢？稍后我们会介绍，先来了解如何制作Bootstrap框架中的弹出框。

## 6.1 弹出框--弹出框的结构

简单回忆一下，在制作**提示框（tooltip）**时，可以使用<button>或者<a>标签元素，而且通过 data- 属性来声明提示框的信息，如下所示：

<button type="button"

class="btnbtn-default"

data-toggle="tooltip"

data-placement="left"

data-original-title="我是一个提示框tooltip">

鼠标移上来

</button>

而弹出框Popover和提示框tooltip相比，**就多了一个content内容**，那么在此使用 **data-content** 来定义弹出框中的内容。同样可以使用<button>或者<a>标签来制作，如下：

<button type="button"

class="btnbtn-default"

data-container="body"

data-placement="bottom"

data-toggle="popover"

data-original-title="Bootstrap弹出框标题"

data-content="Bootstrap弹出框的内容" >

猛击我吧

</button>

<a href="#" class="btnbtn-default"

data-container="body"

data-placement="right"

data-toggle="popover"

title="Bootstrap弹出框标题"

data-content="Bootstrap弹出框的内容">

猛击我吧

</a>

运行效果如下：  
 [](http://img.mukewang.com/541a9f070001fdc206590253.jpg)  
 正如上图所示，弹出框的制作结构和提示框并无太多差别，但样式风格上还是有蛮大的区别。其主要定义了弹出框边框、圆角、背景、阴影以及三角形等样式。

## 6.2 弹出框--触发弹出框的方法

Bootstrap框架中触发弹出框和提示框一样不能直接通过HTML的自定义data属性来触发。需要依赖于JavaScript脚本。  
**最简单的触发方式如下：**

html代码：

<button type="button"

class="btn btn-default"

data-toggle="popover"

data-placement="left"

title="提示框居左"

data-content="我是弹出框的内容">

猛击我吧

</button>

js代码：

$(function(){

$('[data-toggle="popover"]').popover();

});

注意：上面这种方法注意要使用 data- 设置弹出框的属性。

使用JS设置参数：

html代码：

<button type="button"

class="btn btn-default"

id="myPopover">

猛击我吧

</button>

除此之外，也可以单独指定一个元素，在该元素上调用提示框，并且可能通过JavaScript的各种参数来实现，无需定义一些data属性，如：

$(function(){

$('#myPopover').popover({

title:"我是弹出框的标题",

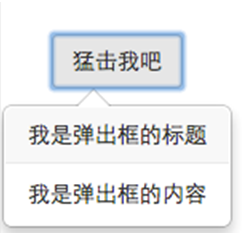
content:"我是弹出框的内容",

placement:"top"

});

});

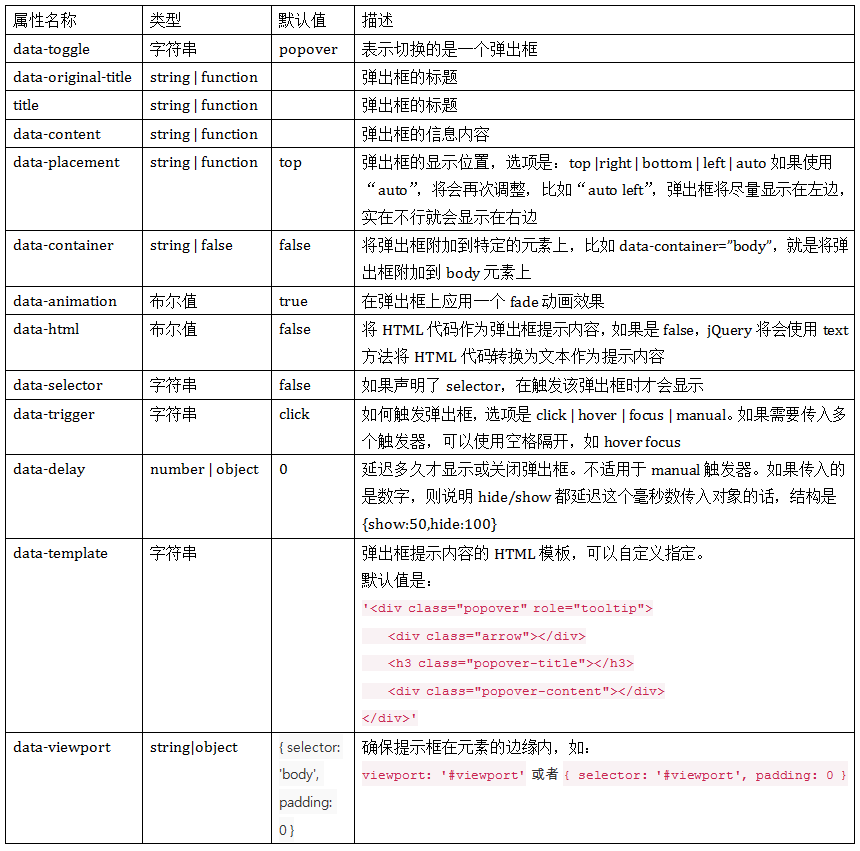
如下面示例所示：

[](http://img.mukewang.com/541aabee00011f3702420233.jpg)

调用popover的时候，options的参数与声明式选择里以data-开头的自定义属性一样。都可以在options里设置，以便制作出各式各样的效果。

## 6.3弹出框--自定义结构属性

同样在弹出框制作时，可以在HTML中定义下表所列的自定义属性：

[](http://img.mukewang.com/54af444c0001dd6a08600851.jpg)

举例：

<button type="button"

class="btn btn-default"

data-toggle="popover"

data-placement="bottom"

title="提示框居左"

data-content="我是弹出框的内容"

data-trigger="hover"

data-dalay="600">

猛击我吧

</button>

# **7 警告框--结构与样式**

在介绍警告框结构之前，我们先来看看警告框插件的效果与警告框的效果在样式风格展示上有何不同：

[IMG_256](http://img.mukewang.com/541b908d000149a806840046.jpg)

上图是“警示框”效果。

[IMG_257](http://img.mukewang.com/541b90ab0001fc0006790043.jpg)

上图“警告框插件”效果。

相比之下，唯一不同的区别正如前面开头所言，警告框插件就是在“警示框组件”基础上添加了一个关闭的按钮，外形上就是一个X。所以其结构只需要在警示框组件的基础上添加一个按钮即可：

<div class="alert " role="alert">

<buttonclass="close" type="button" >&times;</button>

恭喜您操作成功！

</div>

除此差别之外，警告框插件与警示框没有其他的区别

## 7.1 警告框--使用声明式触发警告框

如果通过自定义的HTML属性（声明式）来触发警告框，需要在关闭按钮上设置自定义属性**data-dismiss="alert"**，如下所示：

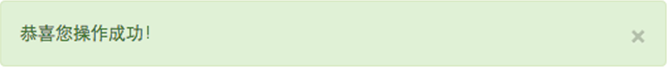
<div class="alert alert-success" role="alert">

<button class="close" data-dismiss="alert" type="button" >&times;</button>

<p>恭喜您操作成功！</p>

</div>

运行效果如下：

[](http://img.mukewang.com/541b91b300018a2b06670067.jpg)

**点击X会关闭整个警告框。**

其实关闭按钮，不一定非要用X号，也可以是普通的按钮元素或者链接元素，只需要保证关闭元素带有自定义属性data-dismiss="alert"即可，如下面示例所示：

<div class="alert alert-warning" role="alert">

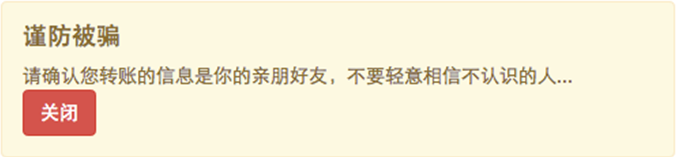
<h4>谨防被骗</h4>

<p>请确认您转账的信息是你的亲朋好友，不要轻意相信不认识的人...</p>

<a href="#" class="btn btn-danger" data-dismiss="alert">关闭</a>

</div>

运行效果如下：

[](http://img.mukewang.com/541b9235000163ae06770157.jpg)

当用户点击“关闭”按钮就可以关闭整个警告框。

## 7.3 警告框--JavaScript触发警告框

除了通过自定义**data-dismiss="alert"**属性来触发警告框关闭之外，还可以通过JavaScript方法。只需要在关闭按钮上绑定一个事件。如下所示：

html代码:

<div class="alert alert-warning" role="alert" id="myAlert">

<h4>谨防被骗</h4>

<p>请确认您转账的信息是你的亲朋好友，不要轻意相信不认识的人...</p>

<button type="button" class="btn btn-danger" id="close">关闭</button>

</div>

通过下面的JavaScript代码来触发：

$(function(){

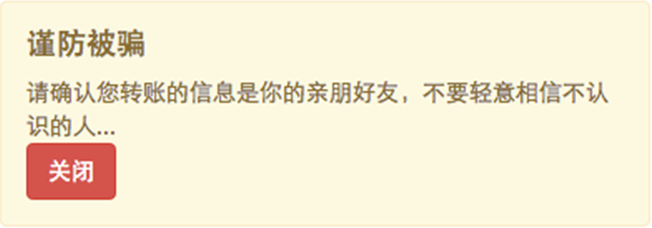
$("#close").on("click",function(){

$(this).alert("close");

});

});

运行效果如下：

[](http://img.mukewang.com/541b9560000135fe06520228.jpg)

# 8 按钮

## 8.1 按钮插件--按钮加载状态

通过按钮可以设计状态提示，当单击按钮时，会显示loading状态信息。例如，点击“加载”按钮，会触发按钮的加载的状态。如下所示：

<button class="btnbtn-primary" data-loading-text="正在加载中,请稍等..." type="button" id="loaddingBtn">加载</button>

通过**data-loading-text**属性定义加载的文本信息，然后通过JavaScript给按钮绑定一个事件，并给按钮添加一个**button("loading")**方法来激活按钮的加载状态行为。如下所示：

$(function(){

$("#loaddingBtn").click(function () {

$(this).button("loading");

});

});

运行效果如下：

点击前：

[IMG_256](http://img.mukewang.com/541b97b400016c6500910057.jpg)

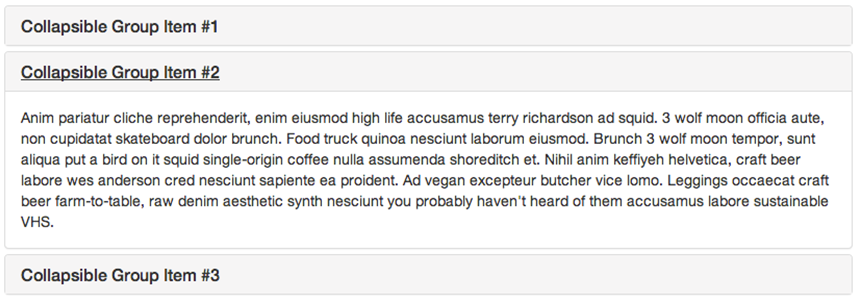
点击后：

[IMG_257](http://img.mukewang.com/541b97d80001794602560057.jpg)

**注意，无法直接通过声明式触发此效果。**

# 9 手风琴（Collapse）

Bootstrap 框架中 Collapse插件(折叠)其实就是我们常见的手风琴效果。点击标题，可以让其对应的内容显示或隐藏。如下图所示：

[](http://img.mukewang.com/54a246f20001eabc08570301.jpg)

## 9.1 手风琴--手风琴结构

手风琴最关键的部分，就是每个标题对应有一个内容，在Bootstrap框架中将这两个部分组合起来称为一个**panel页板**，将多个面板组合在一起，就是一个**面板组合 panel-group，**也就是手风琴的结构。如：

<div class="panel-group" id="accordion">

<div class="panel panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a **data-toggle="collapse"** **data-parent="#accordion"** href="#collapseOne">标题一</a></h4>

</div>

<div id="collapseOne" class="panel-collapse collapse in">

<div class="panel-body">标题一对应的内容</div>

</div>

</div>

<div class="panel panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a data-toggle="collapse" data-parent="#accordion" href="#collapseTwo">标题二</a></h4>

</div>

<div id="collapseTwo" class="panel-collapse collapse">

<div class="panel-body">标题二对应的内容</div>

</div>

</div>

<div class="panel panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a data-toggle="collapse" data-parent="#accordion" href="#collapseThree">标题三</a></h4>

</div>

<div id="collapseThree" class="panel-collapse collapse">

<div class="panel-body">标题三对应的内容</div>

</div>

</div>

</div>

简单点就是一个触发器和一个折叠区：

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

<div id="demo" class="collapse in">折叠区</div>

## 9.2手风琴--声明式触发手风琴（一）

触发手风琴可以通过自定义的**data-toggle** 属性来触发。其中**data-toggle值设置为 collapse，data-target="#折叠区标识符"**。接下来我们来看一个简单的示例：（六步）  
**第一步:**设计一个面板组合，里面有三个折叠区：

<div class="panel-group" id="myAccordion">

<div class="panel panel-accordion panel-default"></div>

<div class="panel panel-accordion panel-default"></div>

<div class="panel panel-accordion panel-default"></div>

</div>

你将看到这样的一个效果：

[](http://img.mukewang.com/541ba1410001aef506260132.jpg)

**第二步**：给面板添加内容，每个面板包括两个部分，第一个是面板标题 **panel-heading，并且在这里面添加标题 panel-title**。**第二个部分是面板内容，也就是折叠区，使用 panel-collapse 样式**。

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

<h4 class="panel-title">标题一</h4>

</div>

<div class="panel-collapse">

<div class="panel-body">折叠区内容...</div>

</div>

</div>

…

此时你将看到下图这样的效果：

[](http://img.mukewang.com/541ba1ad0001dbf006260469.jpg)

**第三步**，为了把标题和内容区捆绑在一起，可以通过锚链接的方法，把标题区域和面板区连在一起：

<div class="panel-group" id="myAccordion">

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a href="#panel1" >标题一</a></h4>

</div>

<div class="panel-collapse" id="panel1">

<div class="panel-body">折叠区内容...</div>

</div>

</div>

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a href="#panel2">标题二</a></h4>

</div>

<div class="panel-collapse" id="panel2">

<div class="panel-body">折叠区内容...</div>

</div>

</div>

......

</div>

**第四步**，控制面板内容区是否可视。在Bootstrap框架中，如果你想让**内容区域不可视**，只需要在 **panel-collapse 样式上添加 collapse**：

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a href="#panel1">标题一</a></h4>

</div>

<div class="panel-collapse collapse" id="panel1">

<div class="panel-body">折叠区内容...</div>

</div>

</div>

此时你会看到效果如下图所示：

[](http://img.mukewang.com/54a24c2b0001108b07880274.jpg)

每个面板的内容区都被隐藏起来了，变成不可视，但有时候希望默认第一个面板内容是可视的，需要怎么办？其实Bootstrap作者早就为大家做了这方面的考虑，你只需要**在collapse基础上再追加 in 样式**：

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a href="#panel1">标题一</a></h4>

</div>

<div class="panel-collapse collapse in" id="panel1">

<div class="panel-body">折叠区内容...</div>

</div>

</div>

效果如下：

[](http://img.mukewang.com/54a24c5300012c3107940387.jpg)

**第五步**，激活手风琴交互行为。要完成交互行为，需要在标题链接中自定义**两个属性**，**一个是data-toggle**，**并且取值为collapse**；**另一个是data-target**，**取值为各个面板内容区的标识符，比如说ID**，在这个例子分别是#panel1、#panel2和#panel3：

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

<h4 class="panel-title"><a href="#panel1" data-toggle="collapse" data-target="#panel1">标题一</a></h4>

</div>

<div class="panel-collapse collapse in" id="panel1">

<div class="panel-body">折叠区内容...</div>

</div>

</div>

到此，实现了单个面板的“显示/隐藏”的切换。但离手风琴效果还略有差距。

**注意：**在这个案例中不加入data-target="#panel1"也可以，因为前面已经有了href="#panel1"，但如是button按钮作为触发器就必须使用data-target="#panel1"语句了。

**第六步**，定义**data-parent属性**，实现点击一个其中一个元素时，关闭所有的折叠区，再打开所单击的区域（如果所单击区域是展示的，则会关闭）。这个**data-parent取值与手风琴面板容器的标识符相匹配**，比如这个例子是指**#myAccordion**：

<div class="panel-group" id="myAccordion">

<div class="panel panel-accordion panel-default">

<div class="panel-heading">

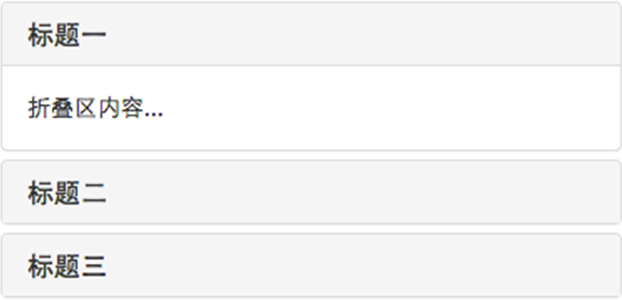
<h4 class="panel-title">

<a href="#panel1" data-toggle="collapse" data-target="#panel1" data-parent="#myAccordion">标题一</a>

</h4>

</div>

…

最终效果如下：[](http://img.mukewang.com/541ba53f0001fd3806220300.jpg)

通过示例，我们可以看出以下几点：

**☑ 使用 panel 的 panel-title 做为触元素，使用panel-body的父元素作为折叠区；**

**☑ 使用一个 panel-group 来包含多个 panel，从而实现手风琴效果；**

**☑ 每个 panel 里的触发元素都要指定data-parent属性；**

**☑ data-parent 属性的值对应 panel-group样式元素的ID或者其他样式标识符；**

**☑ 触发元素需要指定 data-toggle，并且值为 collapse；**

**☑ 触发元素都要指定 data-target属性；**

**☑ data-target属性的值对应 panel-body 的父元素的ID或者其他样式标识符。**

# 10 图片轮播（Carousel）

图片轮播效果在Web中常常能看到，很多人也称之为幻灯片。其主要显示的效果就是多幅图片轮回播放，如下图所示：

[](http://img.mukewang.com/541bc45400010b8506920174.jpg)

上面的轮播效果是6张广告图从右向左播放，鼠标悬停在图片时会暂停播放，如果鼠标悬停或单击右下角圆点时，会显示对应的图片。这种图片轮播效果，在Bootstrap框架中是通过Carousel插件来实现，在下面小节中我们将要介绍的是如何使用Carouse插件实现图片轮播效果。

## 10 .1 图片轮播--轮播图片的设计

一个轮播图片主要包括三个部分：

**☑ 轮播的图片**

**☑ 轮播图片的计数器**

**☑ 轮播图片的控制器**

复杂一点的轮播图片，每个轮播区会带有对应的标题和描述内容。那么在 Bootstrap 框架中，轮播图是如何设计的呢？

**.container-fluid默认会占满屏幕，只有两边有一些间距**

**第一步**：加上.container-fluid这个最大的父容器。设计轮播图片的容器。在 Bootstrap 框架中采用 **carousel** 样式，并且给这个容器定义一个 ID 值，方便后面采用 data 属性来声明触发。

<div id="slidershow" class="carousel"></div>

**第二步**：设计轮播图片计数器。在容器 div.carousel 的内部添加轮播图片计算器，采用 **carousel-indicators** 样式，其主要功能是显示当前图片的播放顺序(有几张图片就放置几个li)，一般采用有顺列表来制作：

<div id="slidershow" class="carousel">

<!-- 设置图片轮播的顺序 -->

<ol class="carousel-indicators">

<li class="active">1</li>

<li>2</li>

<li>3</li>

<li>4</li>

<li>5</li>

...

</ol>

</div>

在 Bootstrap 框架中，轮播图片计数器，都是以圆点向大家呈现。

**第三步：**设计轮播图片播放区。轮播图整个效果中，播放区是最关键的一个区域，这个区域主要用来放置需要轮播的图片。这个区域使用**carousel-inner** 样式来控制，而且其同样放置在**carousel**容器内，并且通过 **item**容器来放置每张轮播的图片：

<div id="slidershow" class="carousel">

<!-- 设置图片轮播的顺序 -->

<ol class="carousel-indicators">

<li class="active">1</li>

…

</ol>

<!-- 设置轮播图片 -->

<div class="carousel-inner">

<div class="item active">

<a href="##"><img src="http://images3.c-ctrip.com/rk/201407/ll580x145.jpg" alt=""></a>

</div>

<div class="item">

<a href="##"><img src="http://images3.c-ctrip.com/dj/201408/zj/zj\_580145.jpg" alt=""></a>

</div>

…

<div class="item">

<a href="##"><img src="http://images3.c-ctrip.com/dj/201408/zqgq\_580145.jpg" alt=""></a>

</div>

</div>

</div>

其主要通过 carousel-inner 来控制其样式呈现。

运行效果如下：

[](http://img.mukewang.com/541bc8dc00016bce06780171.jpg)

上面显示的图片区只实现了图片播放，但很多轮播图片效果，在每个图片上还对应有自己的标题和描述内容。其实 Bootstrap 框架中的 Carousel 也提供类似的效果。只需要在 item 中图片底部添加对应的代码：

<div id="slidershow" class="carousel">

<!-- 设置图片轮播的顺序 -->

<ol class="carousel-indicators">

<li class="active">1</li>

…

</ol>

<!-- 设置轮播图片 -->

<div class="carousel-inner">

<div class="item active">

<a href="##"><img src="http://images3.c-ctrip.com/rk/201407/ll580x145.jpg" alt=""></a>

<!-- 图片对应标题和描述内容 -->

<div class="carousel-caption">

<h3>图片标题</h3>

<p>描述内容...</p>

</div>

</div>

…

</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541bca770001ea1506830170.jpg)

**第四步**：设计轮播图片控制器。很多时候轮播图片还具有一个向前播放和向后播放的控制器。在 **Carousel 中通过 carousel-control样式配合 left 和 right 来实现**。其中left表示向前播放，right表示向后播放。其同样放在carousel容器内：

<div id="slidershow" class="carousel">

<!-- 设置图片轮播的顺序 -->

<ol class="carousel-indicators">

…

</ol>

<!-- 设置轮播图片 -->

<div class="carousel-inner">

…

</div>

<!-- 设置轮播图片控制器 -->

<a class="left carousel-control" href="" >

<span class="glyphicon glyphicon-chevron-left"></span>

</a>

<a class="right carousel-control" href="">

<span class="glyphicon glyphicon-chevron-right"></span>

</a>

</div>

通过两个 a 链接然后在内部定义要显示的小图标，一个是向前，一个是向后。

这两个图标都显示在图片容器的上面（z-index的值大于carousel-inner的）。

最终的效果如下：

[](http://img.mukewang.com/541bd0820001eeca06740168.jpg)

上图展示的就是 Bootstrap 框架中 Carousel 插件实现的图片轮播的效果。只是上例展示的仅是设计效果，并不具备动画效果，接下来我们一起来看如何触发其播放效果。

## 10.2图片轮播--声明式触轮播图的播放

触发轮播图的播放方法也有两种，一种是声明式，另外一种是JavaScript方式。首先来看声明式方法。

声明式方法是通过定义 data 属性来实现，data 属性可以很容易地控制轮播的位置。其主要包括以下几种：

**data-ride 属性：**取值 carousel，并且将其定义在 carousel 上。

**data-target 属性：**取值 carousel 定义的 ID 名或者其他样式识别符，如前面示例所示，取值为“#slidershow”，并且将其定义在轮播图计数器的每个 li 上。

**data-slide 属性：**取值包括 prev，next，prev表示向后滚动，next 表示向前滚动。该属性值同样定义在轮播图控制器的 a 链接上，同时设置控制器 href 值为容器 carousel 的 ID 名或其他样式识别符。

**data-slide-to 属性：**用来传递某个帧的下标，比如 data-slide-to="2"，可以直接跳转到这个指定的帧（下标从0开始计），同样定义在轮播图计数器的每个 li 上。

基于上例，设置了自定义的 data 属性之后如下：

<div id="slidershow" class="carousel" data-ride="carousel">

<!-- 设置图片轮播的顺序 -->

<ol class="carousel-indicators">

<li class="active" data-target="#slidershow" data-slide-to="0"></li>

<li data-target="#slidershow" data-slide-to="1"></li>

<li data-target="#slidershow" data-slide-to="2"></li>

<li data-target="#slidershow" data-slide-to="3"></li>

<li data-target="#slidershow" data-slide-to="4"></li>

<li data-target="#slidershow" data-slide-to="5"></li>

</ol>

<!-- 设置轮播图片 -->

<div class="carousel-inner">

…

</div>

<!-- 设置轮播图片控制器 -->

<a class="left carousel-control" href="#slidershow" role="button" data-slide="prev">

<span class="glyphicon glyphicon-chevron-left"></span>

</a>

<a class="right carousel-control" href="#slidershow" role="button" data-slide="next">

<span class="glyphicon glyphicon-chevron-right"></span>

</a>

</div>

在这里需要注意可以为 #slidershow 层添加 **slide** 样式，使用图片与图片切换效果有平滑感。

<div id="slidershow" class="carousel slide" data-ride="carousel">

...

</div>

这样页面一加载就会有轮播效果了且单击向前向后按钮也可以切换图片了：

[](http://img.mukewang.com/541bd3ca00012cea06720168.jpg)

## 10.3 图片轮播--声明式触轮播图的播放（二）

除了data-ride="carousel"、data-slide、data-slide-to 以外，轮播组件还支持其他三个自定义属性：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名称 | 类型 | 默认值 | 描述 |
| data-interval | number | 5000 | 幻灯片轮换的等待时间（毫秒）。如果为false,轮播将不会自动开始循环 |
| data-pause | string | hover | 默认鼠标悬停留在幻灯片区域即停止播放，离开即开始播放 |
| data-wrap | 布尔值 | true | 轮播是否持续循环 |

如下代码实现“轮播不持续循环”和“轮播时间间隔为1秒”。

<div id="slidershow" class="carousel" data-ride="carousel" data-wrap="false" data-interval="1000">

......

</div>

上面三个属性可以在容器元素上定义，也可以在标示符（或左右控制链接）上定义，但是后者定义的优先级比较高。

## 10.4 图片轮播--JavaScript触发方法

默认情况之下，如果 carousel 容器上定义了 **data-ride="carousel"** 属性，页面加载之后就会自动加载轮播图片切换效果。如果没有定义 **data-ride** 属性，可以通过 JavaScript 方法来触发轮播图片切换。具体使用方法如下：

$(".carousel").carousel();

也可以通过容器的 ID 来指定：

$("#slidershow").carousel();

在 carousel() 方法中可以设置具体的参数，如：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名称 | 类型 | 默认值 | 描述 |
| interval | number | 5000 | 幻灯片轮换的等待时间（毫秒）。如果为false,轮播将不会自动开始循环 |
| pause | string | hover | 默认鼠标悬停留在幻灯片区域即停止播放，离开即开始播放 |
| wrap | 布尔值 | true | 轮播是否持续循环 |

使用时，在初始化插件的时候可以传关相关的参数，如：

$("#slidershow").carousel({

interval: 3000

});

实际上，当我们给carousel()方法配置参数之后，轮播效果就能自动切换。但 Bootstrap 框架中的 carousel 插件还给使用者提供了几种特殊的调用方法，简单说明如下：

* **.carousel("cycle")：从左向右循环播放；**
* **.carousel("pause")：停止循环播放；**
* **.carousel("number")：循环到指定的帧，下标从0开始，类似数组；**
* **.carousel("prev")：返回到上一帧；**
* **.carousel("next")：下一帧**

例如，前面的调用方法，向前和向后两个按钮还无法正常工作，其实可以通过 **.carousel("prev")** 和 **.carousel("next")** 方法让他们能正常工作，代码如下：

$(function(){

$("#slidershow").carousel({

interval:2000

});

$("#slidershow a.left").click(function(){

$(".carousel").carousel("prev");

});

$("#slidershow a.right").click(function(){

$(".carousel").carousel("next");

});

});

# 11 . 固定定位（Affix）

 Affix 一词不好翻译，根据其效果，我将其译为固定定位。Affix 插件是从 Bootstrap V2.1 版本新增的一个插件，其主要功能就是通过插件给某个元素（需要固定的元素）添加或删除 affix 类名，实现元素在浏览器窗口中固定（元素带有 affix 类名固定）和不固定（元素不带有 affix 类名）的效果。

**Affix 效果常见的有以下三种：**

**☑ 顶部固定**

**☑ 侧边栏固定**

**☑ 底部固定**

## 11. 1固定定位--声明式触发固定定位

Affix 插件可以对任何元素进行固定定位，其中比较简单的方法，就是通过自定义属性 data 来触发。其主要包括两个参数：

1、**data-spy**：取值 **affix**，表示元素固定不变的。

2、**data-offset：**整数值，比如 90，表示元素 **top** 和 **bottom** 的值都是 **90px**，其包括两种方式：**data-offset-top**和 **data-offset-bottom**。

**data-offset-top**用来设置元素距离顶部的距离。比如 90，表示元素距离顶部 90px，当用户从顶部向下拖动滚动条，当滚动的距离大于 90px 时，affix 元素不再滚动，就会固定在浏览器窗口顶部。

**data-offset-bottom** 刚好与 data-offset-top 相反。

具体使用如下：

<div data-spy="affix" data-offset="90">affix元素</div>

分开设置 data-offset 值方式：

<div data-spy="affix" data-offset-top="90" data-offset-bottom="150">affix元素</div>

我们来看一个简单的示例：

<nav class="navbar navbar-default" role="navigation">

…

</nav>

<div class="container">

<div class="row">

<div class="col-md-3" id="sidebarMenu">

<ul class="navnav-pills nav-stacked" data-spy="affix" data-offset-top="20">

…

</ul>

</div>

<div class="col-md-9">

…

</div>

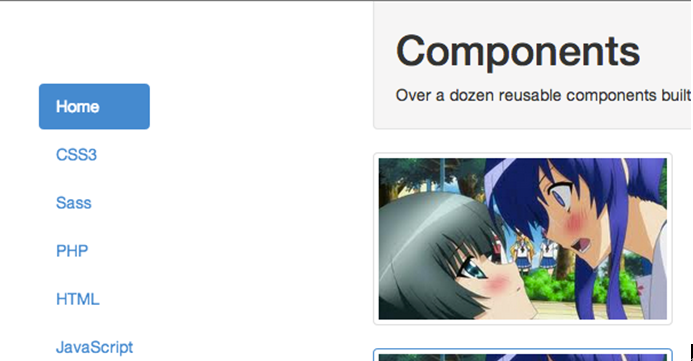
</div>

</div>

**注意，在 body 要声明滚动监控。**

<body data-spy="scroll" data-target="sidebarMenu">

**运行效果如下：**

[](http://img.mukewang.com/541bd80800017c4206930361.jpg)