### 第1章 简介

简单、灵活的用于搭建WEB页面的HTML,CSS,JavaScript的工具集

**Bootstrap 的所有 JavaScript 插件都依赖 jQuery，**因此 jQuery 必须在 Bootstrap 之前引入

## 基本的HTML模板

小伙伴们，上一小节的视频课程已经对Bootstrap提供的模板做了详细的介绍，在这一小节中我们把这个模板代码粘贴过来，以供你们学习查看，你们可以根据实际情况在此基础上进行扩展，只需要确保文件引用顺序一致即可。



**全局样式**

在制作Web页面时，前端人员都习惯为网站设置一个**全局样式（reset.css）**。那么在Bootstrap框架中也不例外。Bootstrap框架的核心是轻量的CSS基础代码库，他并没有一味的重置样式，而是注重各浏览器基础表现，降低开发难度。大部分前端人员都具有归零的思想，但实际你会发现，归零之后的样式在开发过程中会存在着潜在的问题，例如，在全局样式表中将**em**变成一个普通标记，明明应该是斜体，怎么就没效果了呢？

Bootstrap框架在这一部分做了一定的变更，不再一味追求归零，而是更注重重置可能产生问题的样式（如，body,form的margin等），保留和坚持部分浏览器的基础样式，解决部分潜在的问题，提升一些细节的体验，具体说明如下：

* **移除body的margin声明**
* **设置body的背景色为白色**
* **为排版设置了基本的字体、字号和行高**
* **设置全局链接颜色，且当链接处于悬浮“:hover”状态时才会显示下划线样式**

**小伙伴们，你们可以单击查看右侧“style.css”文件，来查看上面所说的全局样式（这个是从下载的boostrap.css中摘取出来的）。**

### 第2章 排版

## 标题（一）

Bootstrap和普通的HTML页面一样，定义标题都是使用标签<h1>到<h6>,只不过Bootstrap覆盖了其默认的样式，使用其在所有浏览器下显示的效果一样，具体定义的规则可以如下表所示：



通过比较可以发现，Bootstrap标题样式进行了以下显著的优化重置：

1、重新设置了**margin-top**和**margin-bottom**的值，  **h1~h3**重置后的值都是**20px**；**h4~h6**重置后的值都是**10px。**  
2、所有标题的行高都是**1.1**（也就是font-size的1.1倍）,而且文本颜色和字体都继承父元素的颜色和字体。  
3、固定不同级别标题字体大小，**h1=36px，h2=30px，h3=24px，h4=18px，h5=14px**和**h6=12px。**

在Bootstrap中为了让非标题元素和标题使用相同的样式，还特意定义了.h1~.h6六个类名



## 标题（二）

除此之外，我们在Web的制作中，常常会碰到在一个标题后面紧跟着一行小的副标题。在Bootstrap中他也考虑了这种排版效果，使用了**<small>**标签来制作副标题。这个副标题具有其自己的一些独特样式：

**1、**行高都是**1**，而且**font-weight**设置了**normal**变成了常规效果（不加粗），同时颜色被设置为**灰色（#999）。**  
**2、**由于**<small>**内的文本字体在**h1~h3**内，其大小都设置为当前字号的**65%；**而在**h4~h6**内的字号都设置为当前字号的**75%；**  
详细代码请参阅**bootstrap.css**文件中第**407行~第443**行。

**段落（正文文本）**

段落是排版中另一个重要元素之一。在Bootstrap中为文本设置了一个全局的文本样式（这里所说的文本是指正文文本）：

1、全局文本字号为**14px(font-size)**。

2、行高为**1.42857143（line-height）**，大约是**20px**(大家看到一串的小数或许会有疑惑，其实他是通过LESS编译器计算出来的，当然Sass也有这样的功能)。

3、颜色为**深灰色（#333）**；

4、字体为**"Helvetica Neue", Helvetica, Arial, sans-serif;（font-family）**，或许这样的字体对我们中文并不太合适，但在实际项目中，大家可以根据自己的需求进行重置，在此我们不做过多阐述，我们回到这里。该设置都定义在**<body>**元素上，由于这几个属性都是**继承属性**，所以Web页面中文本（包括段落p元素）如无重置都会具有这些样式效果。

/\*源码请查看bootstrap.css文件中第274行~280行\*/

body {

font-family: "Helvetica Neue", Helvetica, Arial, sans-serif;

font-size: 14px;

line-height: 1.42857143;

color: #333;

background-color: #fff;

}

另外在Bootstrap中，为了让段落**p**元素之间具有一定的间距，便于用户阅读文本，特意设置了**p**元素的**margin**值（默认情况之下，**p**元素具有一个上下外边距，并且保持一个行高的高度）：  
/\*源码请查看bootstrap.css文件中第467行~469行\*/

p {

margin: 0 0 10px;

}

如果你对CSS预处理器有所了解，那么你完全可以根据Bootstrap提供的预编译版本LESS(或者Sass)进行自定义排版设置。在Bootstrap中，排版设置的默认值都存在**variables.less**文件中(Sass版本存在**\_variables.scss中**)的两个变量：

**LESS版本：**

@font-size-base: 14px; @line-height-base: 1.428571429; // 20/14

**Sass版本：**

$font-size-base: 14px !default; $line-height-base: 1.428571429 !default; // 20/14

第一条语句用于设置**字体大小**，第二条语句用于设置**行高**。系统默认使用这两个值产生整个页面相应的**margin、padding**和**line-height**的值。换句话说，你只需要修改这两个变量的值，然后重新编译，就可以自定义自己的Bootstrap排版样式。（有兴趣的同学可以尝试一下，此处对于LESS或Sass版本运用不做过多阐述）。

## 强调内容

在实际项目中，对于一些重要的文本，希望突出强调的部分都会做另外的样式处理。Bootstrap同样对这部分做了一些轻量级的处理。

如果想让一个段落**p**突出显示，可以通过添加类名“**.lead**”实现，其作用就是增大文本字号，加粗文本，而且对行高和**margin**也做相应的处理。用法如下：

<p>我是普通文本，我的样子长成这样我是普通文本，我的样子长成这样我是普通文本，</p>

<p **class="lead"**>我是特意要突出的文本，我的样子成这样。我是特意要突出的文本，我的样子长成这样。</p>

效果查看最右侧结果窗口。

**“.lead”对应的样式如下：**  
/\*源码查看bootstrap.css文件第470行~480行\*/

.lead {

margin-bottom: 20px;

font-size: 16px;

font-weight: 200;

line-height: 1.4;

}

@media (min-width: 768px) {/\*大中型浏览器字体稍大\*/

.lead {

font-size: 21px;

}

}

除此之外，Bootstrap还通过元素标签:**<small>**、**<strong>**、**<em>**和**<cite>**给文本做突出样式处理。  
/\*源码查看bootstrap.css文件第55行~第58行\*/

b,strong {

font-weight: bold; /\*文本加粗\*/

}

/\*源码查看bootstrap.css文件第481行~第484行\*/

small,.small {

font-size: 85%; /\*标准字体的85%,也就是14px \* 0.85px，差不多12px\*/

}

/\*源码查看bootstrap.css第485行~第487行\*/

cite {

font-style: normal;

}

## 粗体

粗体就是给文本加粗，在普通的元素中我们一般通过font-weight设置为bold关键词给文本加粗。在Bootstrap中，可以使用**<b>**和**<strong>**标签让文本直接加粗。  
/\*源码查看bootstrap.css文件第55行~第58行\*/

b,strong {

font-weight: bold; /\*文本加粗\*/

}

例如，下面的代码使用**<strong>**标签定义了强调文本：

<p>我在学习**<strong>**Bootstrap**</strong>**，我要掌握**<strong>**Bootstrap**</strong>**的所有知识。</p>

## 斜体

在排版中，除了用加粗来强调突出的文本之外，还可以使用斜体。斜体类似于加粗一样，除了可以给元素设置样式**font-style**值为**italic**实现之外，在Bootstrap中还可以通过使用标签**<em>**或**<i>**来实现。  
例如，下面的代码使用了**<em>**和**<i>**标签定义了强调文本：

<p>我在慕课网上跟**<em>**大漠**</em>**一起学习**<i>**Bootstrap**</i>**的使用。我一定要学会**<i>**Bootstrap**</i>**。</p>

**强调相关的类**

在Bootstrap中除了使用标签**<strong>、<em>**等说明正文某些字词、句子的重要性，Bootstrap还定义了一套类名，这里称其为强调类名（类似前面说的“**.lead**”）,这些强调类都是通过颜色来表示强调，具本说明如下：

* .text-muted：提示，使用浅**灰色（#999）**
* .text-primary：主要，使用**蓝色（#428bca）**
* .text-success：成功，使用**浅绿色(#3c763d)**
* .text-info：通知信息，使用**浅蓝色（#31708f）**
* .text-warning：警告，使用**黄色（#8a6d3b）**
* .text-danger：危险，使用**褐色（#a94442）**





## 文本对齐风格

在排版中离不开文本的对齐方式。在CSS中常常使用text-align来实现文本的对齐风格的设置。其中主要有四种风格：

  ☑  左对齐，取值left

  ☑  居中对齐，取值center

  ☑  右对齐，取值right

  ☑  两端对齐，取值justify

为了简化操作，方便使用，Bootstrap通过定义四个类名来控制文本的对齐风格：

  ☑   .text-left：左对齐

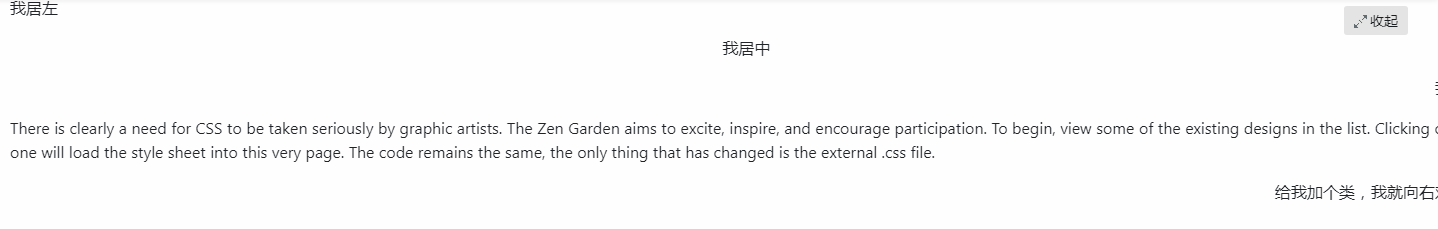
  ☑   .text-center：居中对齐

  ☑   .text-right：右对齐

  ☑   .text-justify：两端对齐

特别声明：目前两端对齐在各浏览器下解析各有不同，特别是应用于中文文本的时候。所以项目中慎用。





## 列表--简介

在HTML文档中，列表结构主要有三种：有序列表、无序列表和定义列表。具体使用的标签说明如下：  
**无序列表**

<ul>

<li>…</li>

</ul>

**有序列表**

<ol>

<li>…</li>

</ol>

**定义列表**

<dl>

<dt>…</dt>

<dd>…</dd>

</dl>

Bootstrap根据平时的使用情形提供了**六种**形式的列表：

   ☑  普通列表

   ☑  有序列表

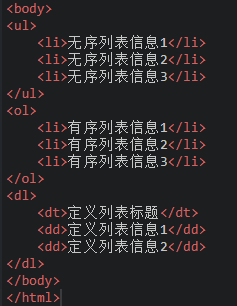
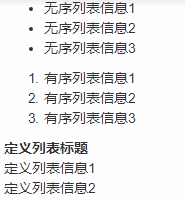
   ☑  去点列表

   ☑  内联列表

   ☑  描述列表

   ☑  水平描述列表

接下来，我们一起来学习Bootstrap提供的这六种列表的使用方法。

## 列表--无序列表、有序列表

**无序列表**

无序列表和有序列表使用方式和我们平时使用的一样（无序列表使用ul，有序列表使用ol标签），在样式方面，Bootstrap只是在此基础上做了一些细微的优化，源码请查看bootstrap.css文件的第569行~第579行：

ul,

ol {

margin-top: 0;

margin-bottom: 10px;

}

ul ul,

ol ul,

ul ol,

ol ol {

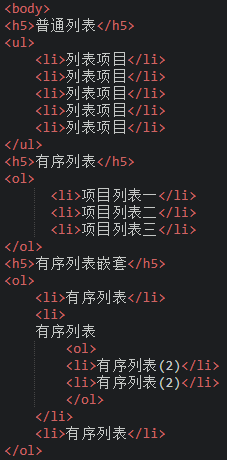
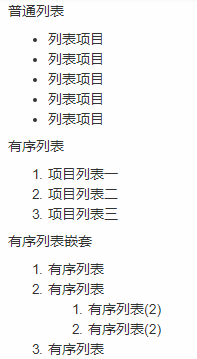
margin-bottom: 0;

}

从源码上我们可以得知，Bootstrap对于列表，只是在**margin**上做了一些调整。我们来看一个简单的示例(右侧代码编辑器10-16行)。

**列表嵌套**

在Bootstrap中列表也是可以嵌套的，如右侧编辑器中的代码就是有序列表（25-35行）。

小伙伴们可以看到，在Bootstrap中默认情况下**无序列表**和**有序列表**是带有项目符号的，但在实际工作中很多时候，我们的列表是不需要这个编号的，比如说用无序列表做导航的时候。Bootstrap为众多开发者考虑的非常周道，通过给无序列表添加一个类名“**.list-unstyled**”,这样就可以去除默认的列表样式的风格。  
/\*源码请查看bootstrap.css文件第580行~第583行\*/

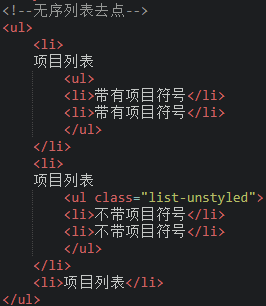
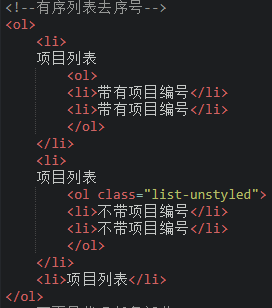
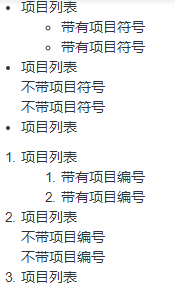
.list-unstyled {

padding-left: 0;

list-style: none;

}

从示例中可以看出，除了项目编号之外，还将列表默认的**左边内距**也**清０**了。

## 列表--内联列表

Bootstrap像去点列表一样，通过添加类名“**.list-inline**”来实现内联列表，简单点说就是**把垂直列表换成水平列表**，**而且去掉项目符号（编号）**，**保持水平显示**。也可以说内联列表就是为制作水平导航而生。  
/\*源码查看bootstrap.css文件第584行~第593行\*/

.list-inline {

padding-left: 0;

margin-left: -5px;

list-style: none;

}

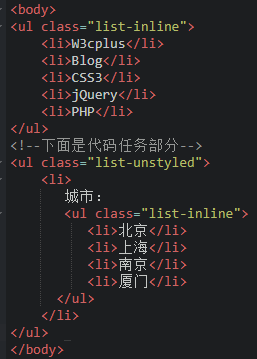
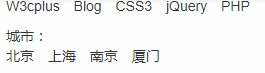
.list-inline > li {

display: inline-block;

padding-right: 5px;

padding-left: 5px;

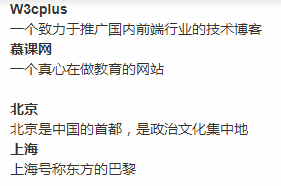
}

## 列表--定义列表

对于定义列表而言，Bootstrap并没有做太多的调整，只是调整了**行间距**，**外边距**和**字体加粗**效果。

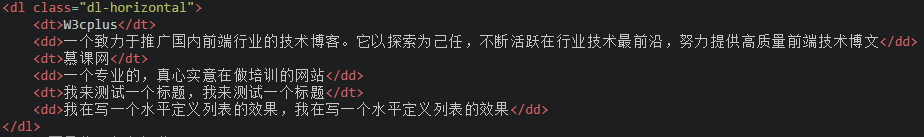


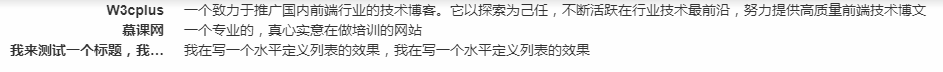


## 列表--水平定义列表

水平定义列表就像内联列表一样，Bootstrap可以给<dl>添加类名“**.dl-horizontal**”给定义列表实现水平显示效果。

此处添加了一个媒体查询。也就是说，只有屏幕**大于768px**的时候，添加类名“**.dl-horizontal**”才具有水平定义列表效果。其实现主要方式：  
**1、将dt设置了一个左浮动，并且设置了一个宽度为160px  
2、将dd设置一个margin-left的值为180px，达到水平的效果  
3、当标题宽度超过160px时，将会显示三个省略号**





## 代码（一）

本节内容相对来说比较简单，一般在个人博客上使用的较为频繁，用于显示代码的风格。在Bootstrap主要提供了**三种代码风格**：  
1、使用**<code></code>**来显示单行内联代码  
2、使用**<pre></pre>**来显示多行块代码  
3、使用**<kbd></kbd>**来显示用户输入代码

在使用代码时，用户可以根据具体的需求来使用不同的类型：  
1、**<code>**：一般是针对于**单个单词或单个句子**的代码  
2、**<pre>**：一般是针对于**多行代码**（也就是成块的代码）  
3、**<kbd>**:一般是表示用**户要通过键盘输入的内容**

虽然不同的类型风格不一样，但其使用方法是类似的。  
**code风格：**

<div>Bootstrap的代码风格有三种：

**<code>&lt;code&gt;</code>**

**<code>&lt;pre&gt;</code>**

**<code>&lt;kbd&gt;</code>**

</div>

**pre风格：**

<div>

**<pre>**

&lt;ul&gt;

&lt;li&gt;...&lt;/li&gt;

&lt;li&gt;...&lt;/li&gt;

&lt;li&gt;...&lt;/li&gt;

&lt;/ul&gt;

**</pre>**

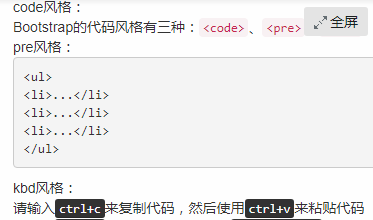
</div>

**kbd风格：**

<div>请输入**<kbd>**ctrl+c**</kbd>**来复制代码，然后使用**<kbd>**ctrl+v**</kbd>**来粘贴代码</div>

不管使用哪种代码风格，在代码中碰到小于号（<）要使用**硬编码**“**&lt;**”来替代，大于号(>)使用“**&gt;**”来替代。而且对于<pre>代码块风格，标签前面留多少个空格，在显示效果中就会留多少个空格。建议在编写HTML标签时，就控制好，例如查看右侧代码编辑器上的15-19行。





## 代码（二）

正如前面所示，<pre>元素一般用于显示大块的代码，并保证原有格式不变。但有时候代码太多，而且不想让其占有太大的页面篇幅，就想控制代码块的大小。Bootstrap也考虑到这一点，你只需要在pre标签上添加类名“**.pre-scrollable**”，就可以控制代码块区域最大高度为**340px**，一旦超出这个高度，就会在**Y轴出现滚动条**。  
/\*源码请查看bootstrap.css第731行~第734行\*/

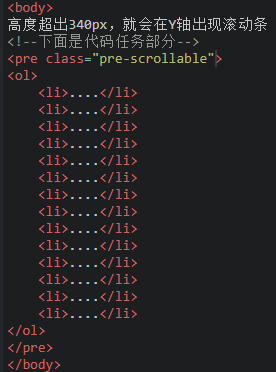
.pre-scrollable {

max-height: 340px;

overflow-y: scroll;

}

友情提示：如果您了解LESS或Sass这样的CSS预定定义处理器，你完全可以通过**code.less**或者**\_code.scss**文件修改样式，然后重新编译，你就可以得到属于自己的代码样式风格。

## 表格

表格是Bootstrap的一个基础组件之一，Bootstrap为表格提供了**1种基础样式**和**4种附加样式**以及**1个支持响应式的表格**。在使用Bootstrap的表格过程中，只需要添加对应的类名就可以得到不同的表格风格，在接下来的内容中，我们会详细介绍Bootstrap的表格使用。

同样的，如果你对CSS预处理器熟悉，你可以使用Bootstrap提供的预处理版本：  
  ☑  LESS版本，对应的文件是 **tables.less**  
  ☑  Sass版本，对应的文件是 **\_tables.scss**

如果你不懂LESS或Sass也不妨，你在bootstrap.css文件中第1402行~第1630行中可以查阅到所有有关于table的样式代码。由于代码太长，此处不一一列举。

刚已经说了，Bootstrap为表格不同的样式风格提供了不同的类名，主要包括：

  ☑  **.table**：基础表格

  ☑  **.table-striped**：斑马线表格

  ☑  **.table-bordered**：带边框的表格

  ☑  **.table-hover**：鼠标悬停高亮的表格

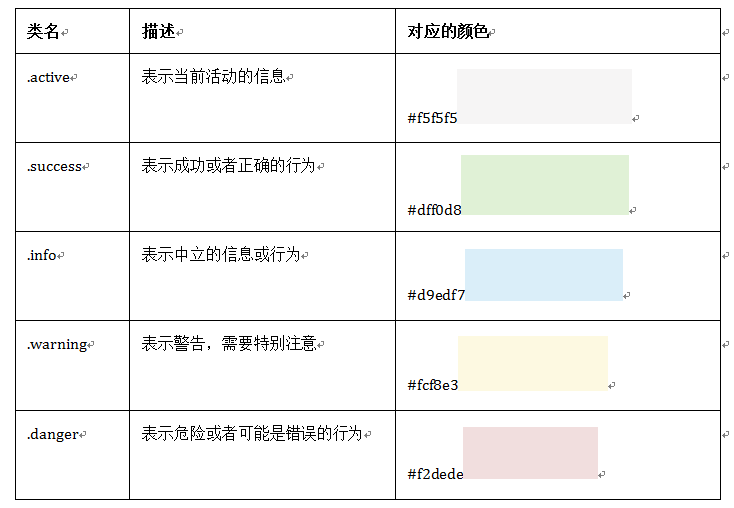
  ☑  **.table-condensed**：紧凑型表格

  ☑  **.table-responsive**：响应式表格

示例见：<https://www.imooc.com/code/2290>

## 表格--表格行的类

Bootstrap还为表格的行元素**<tr>**提供了五种不同的类名，每种类名控制了行的不同背景颜色，具体说明如下表所示：

[](http://img.mukewang.com/53ad213f0001b08807340508.jpg)

其使用非常的简单，只需要在<tr>元素中添加上表对应的类名，就能达到你自己需要的效果：

<tr **class="active"**>

<td>…</td>

</tr>

对应的源码，请查看bootstrap.css文件中第1484行~第1583行。

**特别提示**：除了”.active”之外，其他四个类名和”.table-hover”配合使用时，Bootstrap针对这几种样式也做了相应的悬浮状态的样式设置，所以如果需要给tr元素添加其他颜色样式时，在”.table-hover”表格中也要做相应的调整。

注意要实现悬浮状态，需要在<table>标签上加入table-hover类。

如下代码：

<table class="table-hover">

## 表格--基础表格

大家对表格并不太陌生，但对于Bootstrap中的表格如何使用，或许还有点陌生，接下来的内容，将根据不同的表格类型向大家介绍Bootstrap表格的实际使用方法。  
对表格的结构，跟我们平时使用表格是一样的：

<table>

<thead>

<tr>

<th>表格标题</th>

…

</tr>

</thead>

<tbody>

<tr>

<td>表格单元格</td>

…

</tr>

…

</tbody>

</table>

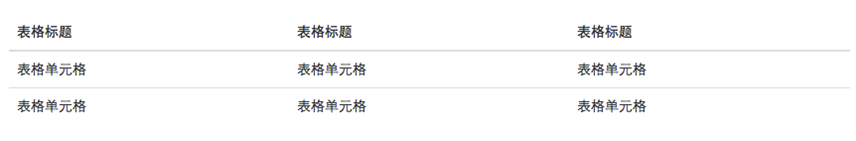
如无特别声明，下面介绍表格类型的时候，结构都是类似上面的代码。  
**基础表格**  
在Bootstrap中，对于基础表格是通过类名“.table”来控制。如果在<table>元素中不添加任何类名，表格是无任何样式效果的。想得到基础表格，我们只需要在<table>元素上添加“.table”类名，就可以得到Bootstrap的基础表格：

<table **class="table"**>

…

</table>

Bootstrap的基础表格，大致长得像下图所示的样子：

[](http://img.mukewang.com/53c617ea0001a48108560141.jpg)

主要源码查看bootstrap.css文件第1402行～第1441行，由于代码太长，此处不一一列举。

**“.table”主要有三个作用：**

**☑  给表格设置了margin-bottom:20px以及设置单元内距**

**☑  在thead底部设置了一个2px的浅灰实线**

**☑  每个单元格顶部设置了一个1px的浅灰实线**

## 表格--斑马线表格

有时候为了让表格更具阅读性，需要将表格制作成类似于斑马线的效果。简单点说就是让表格带有背景条纹效果。在Bootstrap中实现这种表格效果并不困难，只需要在**<table class="table">**的基础上增加类名“**.table-striped**”即可：

<table class="**table table-striped**">

…

</table>

其效果与基础表格相比，仅是在tbody**隔行有一个浅灰色的背景色**。其实现原理也非常的简单，利用CSS3的结构性选择器“:nth-child”来实现，所以对于IE8以及其以下浏览器，没有背景条纹效果。

源码请查看bootstrap.css文件第1465行~第1468行：

.table-striped > tbody > tr:nth-child(odd) > td,

.table-striped > tbody > tr:nth-child(odd) > th {

background-color: #f9f9f9;

}

## 表格--带边框的表格

基础表格仅让表格部分地方有边框，但有时候需要整个表格具有边框效果。Bootstrap出于实际运用，也考虑这种表格效果，即所有单元格具有一条**1px**的边框。  
Bootstrap中带边框的表格使用方法和斑马线表格的使用方法类似，只需要在基础表格<table class="table">基础上添加一个“**.table-bordered**”类名即可：

<table class="table **table-bordered**">

…

</table>

样式如下图所示：

[](http://img.mukewang.com/53c6218300019ab105870211.jpg)

## 表格--鼠标悬浮高亮的表格

当鼠标悬停在表格的行上面有一个高亮的背景色，这样的表格让人看起来就是舒服，时刻告诉用户正在阅读表格哪一行的数据。Bootstrap的确没有让你失望，他也考虑到这种效果，其提供了一个“**.table-hover**”类名来实现这种表格效果。  
鼠标悬停高亮的表格使用也简单，仅需要<table class="table">元素上添加类名“**table-hover**”即可：

<table class="**table table-hover**">

…

</table>

效果图如下：



从效果图中可以看出，当你鼠标悬浮在某一单元格上时，单元格所在行的背景色都会变成**浅灰色**。

鼠标悬浮高亮的效果主要是通过“hover”事件来实现，设置了“**tr:hover**”时的th、td的背景色为新颜色。  
其源码请查看bootstrap.css文件中第1469行~第1472行：

.table-hover > tbody > tr:hover > td,

.table-hover > tbody > tr:hover > th {

background-color: #f5f5f5;

}

**注：**其实，鼠标悬浮高亮表格，可以和Bootstrap其他表格混合使用。简单点说，只要你想让你的表格具备悬浮高亮效果，你只要给这个表格添加“**table-hover**”类名就好了。例如，将前面介绍的几种表格结合使用：

<table class="**table table-striped table-bordered** **table-hover**">

…

</table>

## 表格--紧凑型表格

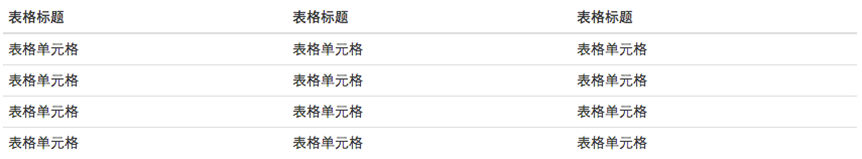
何谓紧凑型表格，简单理解，就是**单元格没内距**或者**内距较其他表格的内距更小**。换句话说，要实现紧凑型表格只需要重置表格单元格的内距**padding**的值。那么在Bootstrap中，通过类名“**table-condensed**”重置了单元格内距值。  
紧凑型表格的运用，也只是需要在<table class="table">基础上添加类名“**table-condensed**”：

<table class="**table table-condensed**">

…

</table>

效果图如下：

[](http://img.mukewang.com/53c62647000158e708620159.jpg)

从上面效果图可以看出，Bootstrap中紧凑型的表格与基础表格差别不大，因为只是将单元格的内距由**8px调至5px**。

正如上一小节中悬浮高亮表格中所讲解的，在Bootstrap中制作表格中，可以将上面几种表格样式结合在一起使用，从而制作出更为精美的表格。结合的方法也很简单，就是在<table class="table">基础上添加你需要的表格样式类型。

另外从上面的示例中大家可能也发现了，不管制作哪种表格都离不开类名“table”。所以大家在使用Bootstrap表格时，千万注意，你的<table>元素中一定不能缺少类名“table”。

## 表格--响应式表格

随着各种手持设备的出现，要想让你的Web页面适合千罗万像的设备浏览，**响应式设计**的呼声越来越高。在Bootstrap中也为表格提供了响应式的效果，将其称为**响应式表格**。  
Bootstrap提供了一个容器，并且此容器设置类名“.table-responsive”,此容器就具有响应式效果，然后将<table class="table">置于这个容器当中，这样表格也就具有响应式效果。  
Bootstrap中响应式表格效果表现为：当你的浏览器可视区域小于**768px**时，表格底部会出现水平滚动条。当你的浏览器可视区域大于768px时，表格底部水平滚动条就会消失。示例如下：

<div class="table-responsive">

<table class="table table-bordered">

…

</table>

</div>

运行效果如下：

[](http://img.mukewang.com/53ad2eab0001c55105540130.jpg)

（宽屏效果）

[](http://img.mukewang.com/53ad2f540001847402420197.jpg)

（窄屏效果）

### 第3章 表单

本章介绍Bootstrap框架核心部分之一：表单，通过本章节的学习，可以轻松的使用Bootstrap框架制作出适合自己需求的表单。

## 基础表单

表单主要功能是用来与用户做交流的一个网页控件，良好的表单设计能够让网页与用户更好的沟通。表单中常见的元素主要包括：**文本输入框**、**下拉选择框、单选按钮、复选按钮**、**文本域**和**按钮**等。其中每个控件所起的作用都各不相同，而且不同的浏览器对表单控件渲染的风格都各有不同。

同样，表单也是Bootstrap框架中的核心内容，下面向大家介绍Bootstrap框架中表单的制作。

**表单源码查询**

Bootstrap框架的表单，其源码占据了大量的代码，同样的，根据不同的Bootstrap版本，你可以轻松获取相应的源码：

   ☑   LESS版本：对应源文件 **forms.less**

   ☑ ​  Sass版本：对应源文件 **\_forms.scss**

编译后的Bootstrap版本，可以查阅bootstrap.css文件第1630行~第1991行

不过在样式表中，还对表单做了一些初始化，详细代码可以查阅bootstrap.css文件第110行~第178行。

对于基础表单，Bootstrap并未对其做太多的定制性效果设计，仅仅对表单内的**fieldset**、**legend**、**label**标签进行了定制。如：

/\*源码请查阅bootstrap.css文件第1631行~第1652行\*/

主要将这些元素的**margin**、**padding**和**border**等进行了细化设置。

当然表单除了这几个元素之外，还有**input**、**select**、**textarea**等元素，在Bootstrap框架中，通过定制了一个类名`**form-control**`，也就是说，如果这几个元素使用了类名“form-control”，将会实现一些设计上的定制效果。

**1、宽度变成了100%**

**2、设置了一个浅灰色（#ccc）的边框**

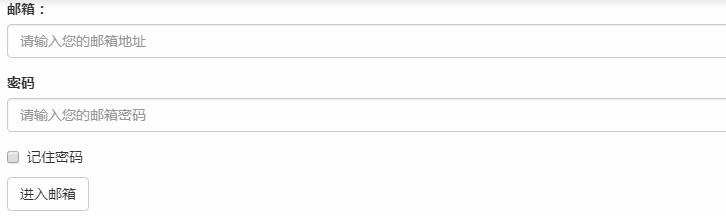
**3、具有4px的圆角**

**4、设置阴影效果，并且元素得到焦点之时，阴影和边框效果会有所变化**

**5、设置了placeholder的颜色为#999**

详细请查阅bootstrap.css文件第1690行~第1732行。





## 水平表单

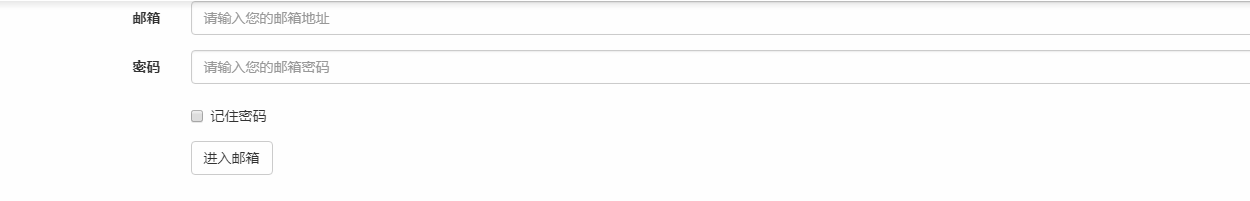
Bootstrap框架默认的表单是**垂直显示风格**，但很多时候我们需要的**水平表单风格**（标签居左，表单控件居右）见下图。

[](http://img.mukewang.com/53d07cb5000111c403540091.jpg)

在Bootstrap框架中要实现水平表单效果，必须满足以下两个条件：  
**1、在<form>元素是使用类名“form-horizontal”。  
2、配合Bootstrap框架的网格系统。（网格布局会在以后的章节中详细讲解）**

在<form>元素上使用类名“form-horizontal”主要有以下几个作用：  
**1、设置表单控件padding和margin值。  
2、改变“form-group”的表现形式，类似于网格系统的“row”。**





## 内联表单

有时候我们需要将表单的控件都在一行内显示，类似这样的：

[http://img.mukewang.com/53b2532a000107b003190032.jpg](http://img.mukewang.com/53b2532a000107b003190032.jpg)

在Bootstrap框架中实现这样的表单效果是轻而易举的，你只需要在<form>元素中添加类名“form-inline”即可。  
内联表单实现原理非常简单，欲将表单控件在一行显示，就需要将表单控件设置成内联块元素（display:inline-block）。  
/\*源码请查阅bootstrap.css文件第1928行~第1962行\*/

如果你要在input前面添加一个label标签时，会导致input换行显示。如果你必须添加这样的一个label标签，并且不想让input换行，你需要将label标签也放在容器“form-group”中，如：

<div **class="form-group"**>

<label class="sr-only" for="exampleInputEmail2">Email address</label>

</div>

<div **class="form-group"**>

<inputtype="email" class="form-control" id="exampleInputEmail2" placeholder="Enter email">

</div>





在label标签运用了一个类名“sr-only”，标签没显示就是这个样式将标签隐藏了

那么Bootstrap为什么要这么做呢？这样不是多此一举吗？其实不是的，如果没有为输入控件设置label标签，**屏幕阅读器**将无法正确识别。这也是Bootstrap框架另一个优点之处，为残障人员进行了一定的考虑。

删除label标签中的class=“sr-only“后



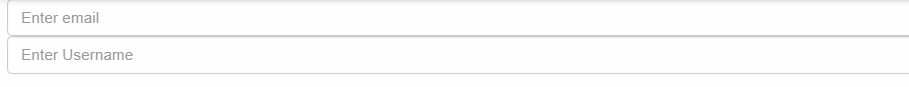
## 表单控件(输入框input)

每一个表单都是由表单控件组成。离开了控件，表单就失去了意义。接下来的我们简单的来了解Bootstrap框架中表单控件的相关知识。

**单行输入框**，常见的文本输入框，也就是**input**的**type**属性值为**text**。在Bootstrap中使用input时也必须添加type类型，如果没有指定type类型，将无法得到正确的样式，因为Bootstrap框架都是通过**input[type=“?”]**(其中?号代表type类型，比如说text类型，对应的是**input[type=“text”]**)的形式来定义样式的。

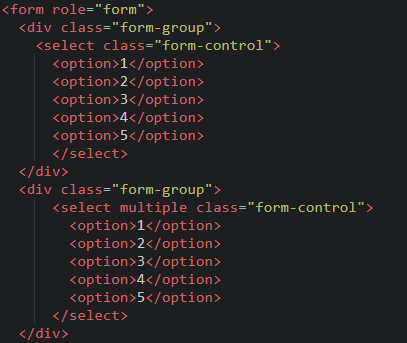
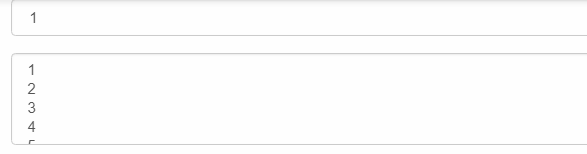
为了让控件在各种表单风格中样式不出错，需要添加类名“**form-control**”





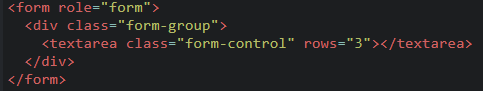
## 表单控件(下拉选择框select)

Bootstrap框架中的下拉选择框使用和原始的一致，多行选择设置**multiple**属性的值为**multiple**。Bootstrap框架会为这些元素提供统一的样式风格。如：

## 表单控件(文本域textarea)

文本域和原始使用方法一样，设置**rows**可定义其高度，设置**cols**可以设置其宽度。但如果**textarea**元素中添加了类名“**form-control**”类名，则无需设置cols属性。因为Bootstrap框架中的“form-control”样式的表单控件宽度为**100%**或**auto**。





## 表单控件(复选框checkbox和单选择按钮radio)

Bootstrap框架中checkbox和radio有点特殊，Bootstrap针对他们做了一些特殊化处理，主要是checkbox和radio与label标签配合使用会出现一些小问题（最头痛的是对齐问题）。使用Bootstrap框架，开发人员无需考虑太多，只需要按照下面的方法使用即可。





从上面的示例，我们可以得知：  
1、不管是checkbox还是radio都使用label包起来了  
2、checkbox连同label标签放置在一个名为“.checkbox”的容器内  
3、radio连同label标签放置在一个名为“.radio”的容器内  
在Bootstrap框架中，主要借助“.checkbox”和“.radio”样式，来处理复选框、单选按钮与标签的对齐方式。

## 表单控件(复选框和单选按钮水平排列)

有时候，为了布局的需要，将复选框和单选按钮需要水平排列。Bootstrap框架也做了这方面的考虑：  
**1、如果checkbox需要水平排列，只需要在label标签上添加类名“checkbox-inline”  
2、如果radio需要水平排列，只需要在label标签上添加类名“radio-inline”**





## 表单控件(按钮)

按钮也是表单重要控件之一,制作按钮通常使用下面代码来实现：

☑  **input[type=“submit”]**

☑  **input[type=“button”]**

☑  **input[type=“reset”]**

☑  **<button>**

在Bootstrap框架中的按钮都是采用<button>来实现。

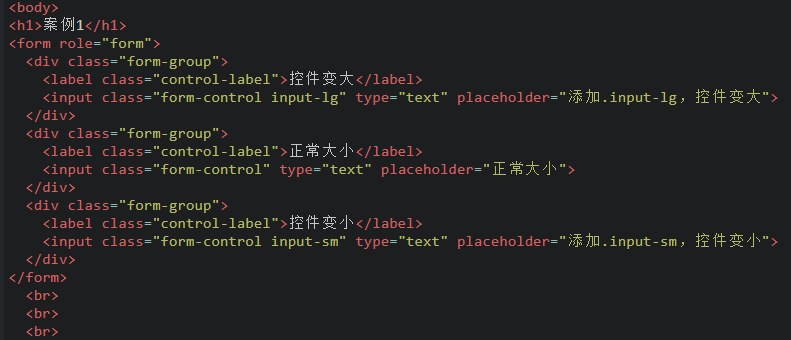
有关于Bootstrap中按钮如何制作，在这里不做过多阐述，因为按钮也是Bootstrap框架中核心部分之一，后面我们专门有一节内容来介绍Bootstrap的按钮。

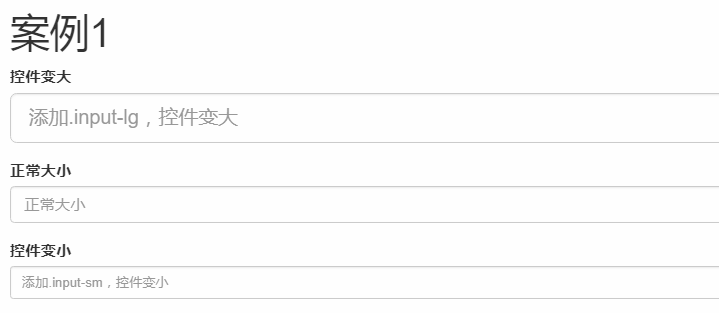


## 表单控件大小

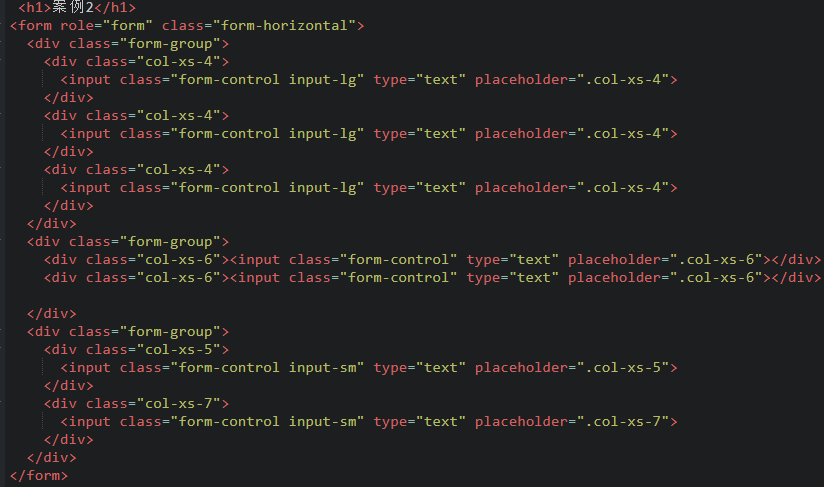
前面看到的表单控件都正常的大小。可以通过设置控件的**height，line-height，padding**和**font-size**等属性来实现控件的高度设置。不过Bootstrap框架还提供了两个不同的类名，用来控制表单控件的高度。这两个类名是：  
**1、input-sm:让控件比正常大小更小  
2、input-lg:让控件比正常大小更大**

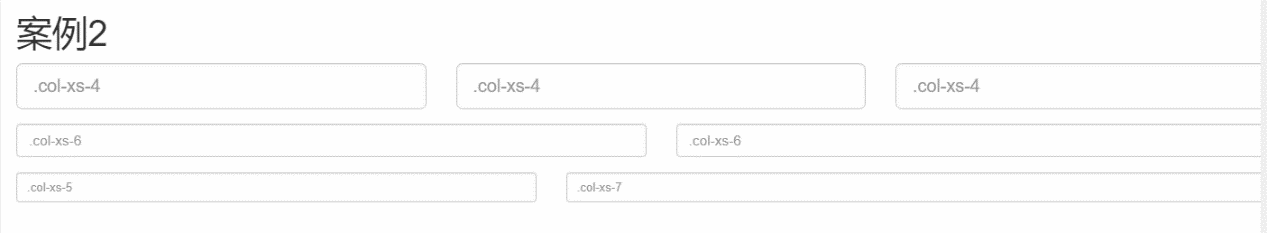
这两个类适用于表单中的**input**，**textarea**和**select**控件





不管是“input-sm”还是“input-lg”仅对控件高度做了处理。但往往很多时候，我们需要控件宽度也要做一定的变化处理。这个时候就要借住Bootstrap框架的**网格系统**。所以你要控制表单宽度，可以像下面这样使用





前面介绍水平表单时说过，如果表单使用了类名“form-horizontal”，其中“form-group”就相当于网格系统中的“row”。换句话说，如果没有这样做，要通过网格系统来控制表单控件宽度，就需要这样使用：

<div class="row">

<div class="col-xs-4">

<input class="form-control input-lg" type="text" placeholder=".col-xs-4">

</div>

<div class="col-xs-4">

<input class="form-control input-lg" type="text" placeholder=".col-xs-4">

</div>

<div class="col-xs-4">

<input class="form-control input-lg" type="text" placeholder=".col-xs-4">

</div>

</div>

## 3-11表单控件状态(焦点状态)

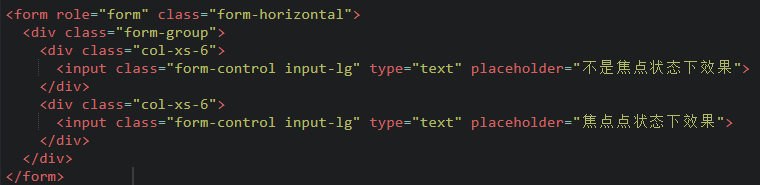
表单主要用来与用户沟通，好的表单就能更好的与用户进行沟通，而好的表单一定离不开表单的控件状态。

**表单状态的作用：**

每一种状态都能给用户传递不同的信息，比如表单有焦点的状态可以告诉用户可以输入或选择东西，禁用状态可以告诉用户不可以输入或选择东西，还有就是表单控件验证状态，可以告诉用户的操作是否正确等。那么在Bootstrap框架中的表单控件也具备这些状态。

焦点状态是通过伪类“:focus”来实现。Bootstrap框架中表单控件的焦点状态删除了**outline**的默认样式，重新添加阴影效果。

从源码中我们可以看出，要让控件在焦点状态下有上面样式效果，需要给控件添加类名“form-control”：



[http://img.mukewang.com/53b271e700018f8a02780042.jpg](http://img.mukewang.com/53b271e700018f8a02780042.jpg)

（鼠标单击输入框，使其获得焦点就可以看到加入蓝色边框效果）

在Bootstrap框架中，**file**、**radio**和**checkbox**控件在焦点状态下的效果也与普通的input控件不太一样，主要是因为Bootstrap对他们做了一些特殊处理：

## 3-12表单控件状态(禁用状态)

Bootstrap框架的表单控件的禁用状态和普通的表单禁用状态实现方法是一样的，在相应的表单控件上添加属性“disabled”。和其他表单的禁用状态不同的是，Bootstrap框架做了一些样式风格的处理：

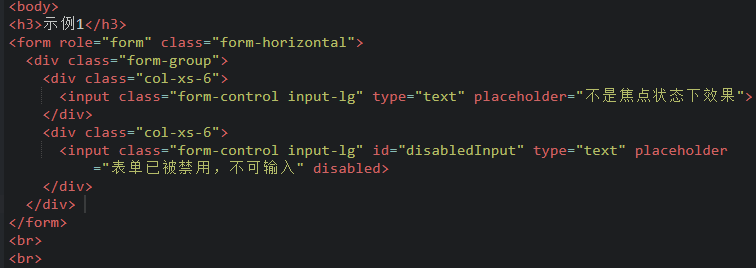
使用方法为：只需要在需要禁用的表单控件上加上“disabled”即可：

<input class="form-control" type="text" placeholder="表单已禁用，不能输入" disabled>

运行效果如下或查看右侧结果窗口：

[http://img.mukewang.com/53b27abf00013e5f03360048.jpg](http://img.mukewang.com/53b27abf00013e5f03360048.jpg)

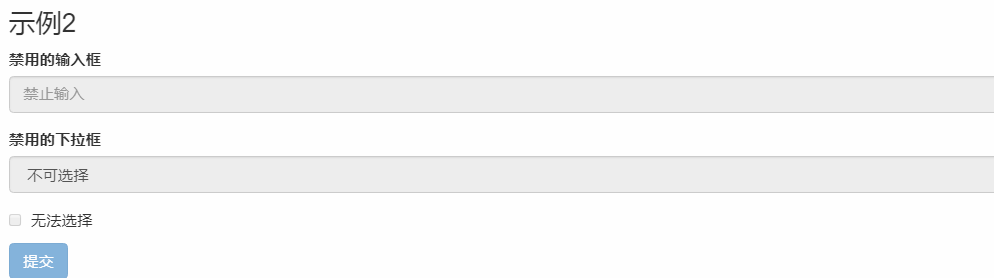
在使用了“form-control”的表单控件中，样式设置了禁用表单背景色为灰色，而且手型变成了不准输入的形状。如果控件中不使用类名“form-control”，禁用的控件只会有一个不准输入的手型出来。





在Bootstrap框架中，如果fieldset设置了disabled属性，整个域都将处于被禁用状态。





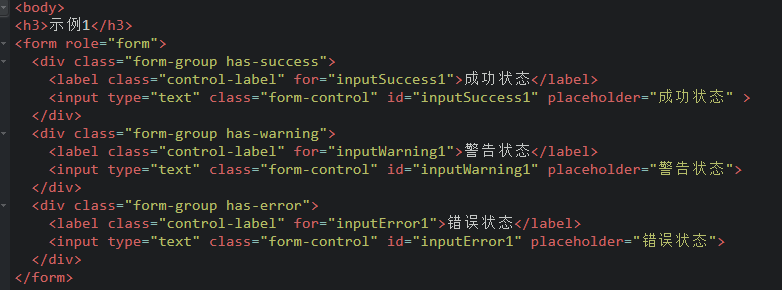
对于整个禁用的域中，如果legend中有输入框的话，这个**输入框是无法被禁用**的。我们一起来验证一下：





## 3-13表单控件状态(验证状态)

在制作表单时，不免要做表单验证。同样也需要提供验证状态样式，在Bootstrap框架中同样提供这几种效果。  
1、.has-warning:警告状态（黄色）  
2、.has-error：错误状态（红色）  
3、.has-success：成功状态（绿色）  
使用的时候只需要在form-group容器上对应添加状态类名。

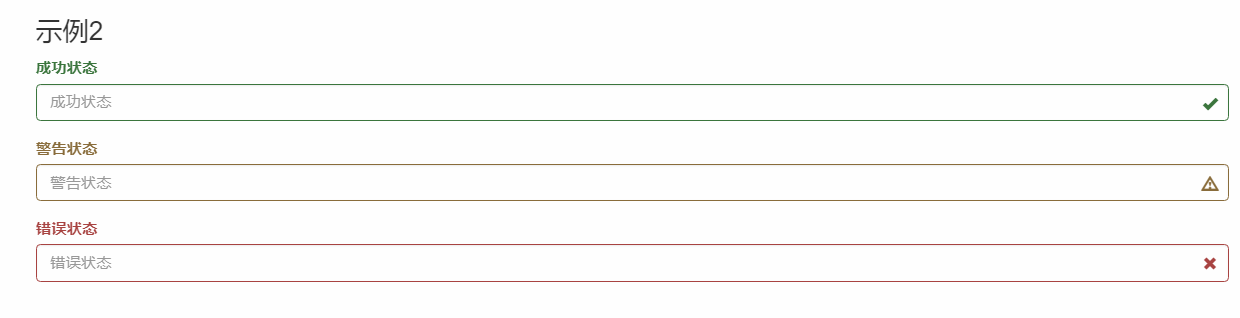




从效果可以看出，三种状态下效果都是一样的，只是颜色不一样而以。

其他两种状态省略源码不在此展示。  
很多时候，在表单验证的时候，不同的状态会提供不同的 icon，比如成功是一个对号（√），错误是一个叉号（×）等。在Bootstrap框中也提供了这样的效果。如果你想让表单在对应的状态下显示 icon 出来，只需要在对应的状态下添加类名“has-feedback”。请注意，此类名要与“has-error”、“has-warning”和“has-success”在一起：





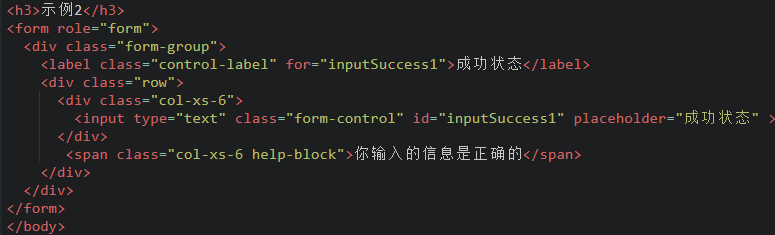
## 3-14表单提示信息

平常在制作表单验证时，要提供不同的提示信息。在Bootstrap框架中也提供了这样的效果。使用了一个"help-block"样式，将提示信息以块状显示，并且显示在控件底部。





在Bootstrap V2.x版本中还提供了一个行内提示信息，其使用了类名“help-inline”。一般让提示信息显示在控件的后面，也就是同一水平显示。如果你想在BootstrapV3.x版本也有这样的效果，你可以添加这段代码. 如果你不想为bootstrap.css增加自己的代码，而且设计又有这种样的需求，那么只能借助于Bootstrap的网格系统。(网格系统在后面的章节中会详细讲解)





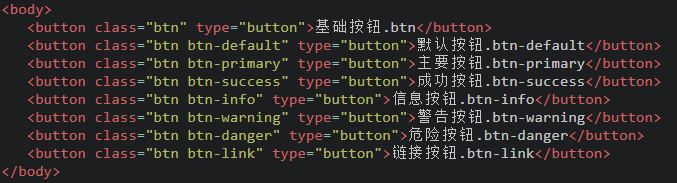
结束语：有关于Bootstrap框架中表单的运用除了按钮部分，到此就算是介绍完了。如果你觉得这样的表单效果并不是你需要的，你完全可以通过**forms.less**或者**\_forms.scss**文件进行定制，然后重新编译就可以得到你需要的表单效果。在接下来的一节中，我们Bootstrap框架中另一个核心内容——**按钮**。

**3-15按钮**

按钮也是Bootstrap框架核心内容之一。因为按钮是Web制作中不可缺少的东西。而且不同的Web页面具有不同的按钮风格，甚至说同一个Web网站或应用程序具有多种按钮风格，比如说不同的按钮颜色、大小和状态等。那么Bootstrap框架也考虑了这些因素，接下来的内容我们一起来探讨Bootstrap框架中的另一核心部分内容——按钮。

Bootstrap框架的按钮也是一个独立部分，我们同样在不同的版本之中能找到对应的代码：

1. **LESS版本**：查看源文件buttons.less
2. **Sass版本**：查看源文件\_buttons.scss
3. **已编译版本：**查看源文件bootstrap.css文件第1992行～第2353行





## 3-17默认按钮

Bootstrap框架首先通过基础类名“.btn”定义了一个基础的按钮风格，然后通过“.btn-default”定义了一个默认的按钮风格。默认按钮的风格就是在基础按钮的风格的基础上修改了按钮的**背景颜色**、**边框颜色**和**文本颜色**。  
/\*源码请查阅bootstrap.css文件第2040行～第2044行\*/

.btn-default {

color: #333;

background-color: #fff;

border-color: #ccc;

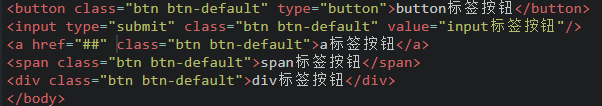
}

使用**默认按钮风格**也非常的简单，只需要在基础按钮“btn”的基础上增加类名“btn-default”即可：

<button class="btn btn-default" type="button">默认按钮</button>

## 3-18多标签支持

一般制作按钮除了使用<button>标签元素之外，还可以使用<input type="submit">和<a>标签等。同样，在Bootstrap框架中制作按钮时，除了刚才所说的这些标签元素之外，还可以使用在其他的标签元素上，唯一需要注意的是，要在制作按钮的标签元素上添加类名“btn”。如果不添加是不会有任何按钮效果。





注意：虽然在Bootstrap框架中使用任何标签元素都可以实现按钮风格，但个人并不建议这样使用，为了避免浏览器兼容性问题，个人强烈建议使用**button**或**a**标签来制作按钮。

## 3-19定制风格

在介绍按钮开篇就说过，Web页面可能会有不同的按钮风格。那么在Bootstrap框架也考虑了。在Bootstrap框架中除了默认的按钮风格之外，还有其他六种按钮风格，每种风格的其实都一样，不同之处就是按钮的背景颜色、边框颜色和文本颜色。

在Bootstrap框架中不同的按钮风格都是通过不同的类名来实现，在使用过程中，开发者只需要选择不同的类名即可：

[](http://img.mukewang.com/53b367bd0001d59c07530312.jpg)

[](http://img.mukewang.com/53b367d10001846a08020810.jpg)

使用起来就很简单，就像前面介绍的默认按钮一样的使用方法，只需要在基础按钮“.btn”基础上追加**对应的类名**，就可以得到需要的按钮风格。如：

<button class="**btn**" type="button">基础按钮.btn</button>

<button class="**btn btn-default**" type="button">默认按钮.btn-default</button>

<button class="**btn btn-primary**" type="button">默认按钮.btn- **primary**</button>

<button class="**btn btn-success**" type="button">成功按钮.btn-success</button>

<button class="**btn btn-info**" type="button">信息按钮.btn-info</button>

<button class="**btn btn-warning**" type="button">警告按钮.btn-warning</button>

<button class="**btn btn-danger**" type="button">危险按钮.btn-danger</button>

<button class="**btn btn-link**" type="button">链接按钮.btn-link</button>

运行效果如下或查看右侧结果窗口：



## 3-20按钮大小

上一节介绍了按钮的定制风格，也就是如何实现Web页面中多种风格按钮。在Bootstrap框架中，对于按钮的大小，也是可以定制的。类似于input一样，通过在基础按钮“.btn”的基础上追加类名来控制按钮的大小。

在Bootstrap框架中提供了三个类名来控制按钮大小：

[](http://img.mukewang.com/53b36a7600014af106910605.jpg)

从上表中不难发现，在Bootstrap框架中控制按钮的大小都是通过修改按钮的**padding**、**line-height**、**font-size**和**border-radius**几个属性。

在实际使用中，这几个类名可以配合按钮中其他颜色类名一起使用，但唯一一点不能缺少“.btn”类名：





## 3-21块状按钮

从前面几节的内容中，大家可能发现了，每个示例中的按钮宽度都是依靠按钮文本和padding的值来决定。但有时候在制作按钮的时候需要按钮宽度充满整个父容器**（width:100%）**，特别是在移动端的制作中。那么前面的方法我们都无法很好的实现，除非重新定义按钮的宽度。其实在Bootstrap中并不需要这样做，Bootstrap框架中提供了一个类名“btn-block”。按钮使用这个类名就可以让按钮充满整个容器，并且这个按钮不会有任何的**padding**和**margin**值。在实际当中，常把这种按钮称为**块状按钮**

使用方法和前面的类似，只需要在原按钮类名上添加“.btn-block”类名，当然“.btn”类名是不可或缺的：



## 3-22按钮状态——活动状态

Bootstrap框架针对按钮的状态做了一些特殊处理。在Bootstrap框架中针对按钮的状态效果主要分为两种：**活动状态**和**禁用状态**。

Bootstrap按钮的活动状态主要包括按钮的**悬浮状态(:hover)**，**点击状态(:active)**和**焦点状态（:focus）**几种。而且不同风格下的按钮都具有这几种状态效果，只是颜色做了一定的调整.

当按钮处理正在点击状态（也就是鼠标按下的未松开的状态），对于<button>元素是通过“:active”伪类实现，而对于<a>这样的标签元素则是通过添加类名“.active”来实现。

## 3-23按钮状态——禁用状态

和input等表单控件一样，在Bootstrap框架的按钮中也具有禁用状态的设置。禁用状态与其他状态按钮相比，就是背景颜色的透明度做了一定的处理，opcity的值从100%调整为65%。

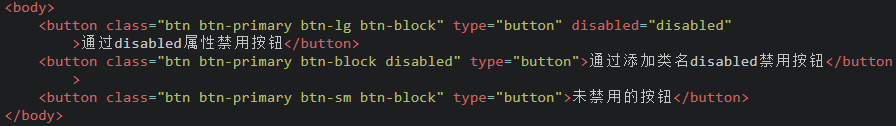
**在Bootstrap框架中，要禁用按钮有两种实现方式：**

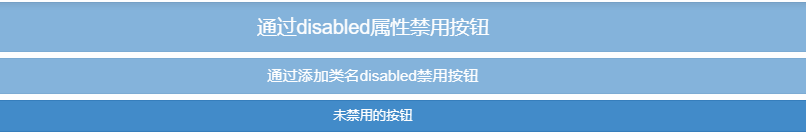
**方法1：**在标签中添加disabled属性

**方法2：**在元素标签中添加类名“disabled”

**两者的主要区别是：**

“.disabled”样式不会禁止按钮的默认行为，比如说提交和重置行为等。如果想要让这样的禁用按钮也能禁止按钮的默认行为，则需要通过JavaScript这样的语言来处理。对于<a>标签也存在类似问题，如果通过类名“.disable”来禁用按钮，其链接行为是无法禁止。而在元素标签中添加“disabled”属性的方法是可以禁止元素的默认行为的。





## 3-24图像

图像在网页制作中也是常要用到的元素，在Bootstrap框架中对于图像的样式风格提供以下几种风格：

**1、img-responsive：**响应式图片，主要针对于响应式设计  
**2、img-rounded：**圆角图片  
**3、img-circle：**圆形图片  
**4、img-thumbnail：**缩略图片

**使用方法：**

使用方法非常简单，只需要在**<img>**标签上添加对应的类名





**设置图片大小：**

由于样式没有对图片做大小上的样式限制，所以在实际使用的时候，需要通过其他的方式来处理图片大小。比如说控制图片容器大小。（注意不可以通过css样式直接修改img图片的大小，这样操作就不响应了）

**注意：**

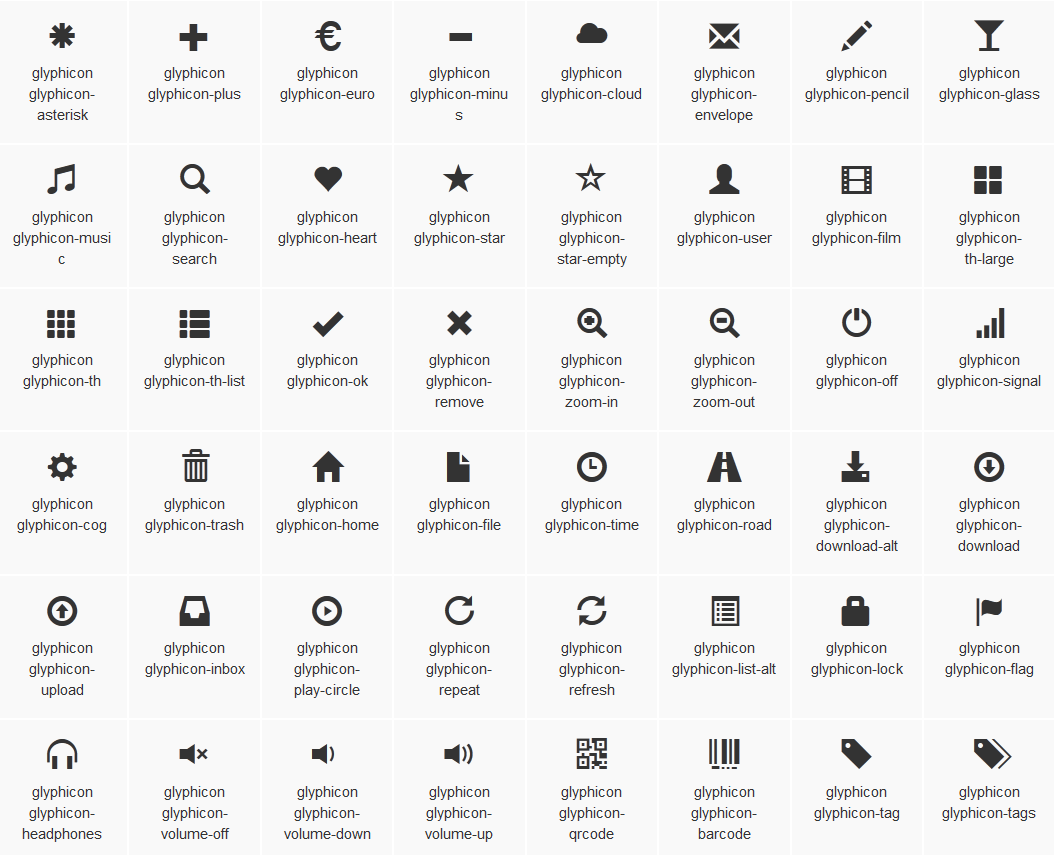
对于圆角图片和圆形图片效果，因为是使用了CSS3的圆角样式来实现的，所以注意对于IE8以及其以下版本不支持，是没有圆角效果的。

Bootstrap框架为了大家更好的维护图像的样式，同样将这部分样式独立出来：  
1、LESS版本，可以查阅scaffolding.less  
2、Sass版本，可以查阅\_scaffolding.scss  
大家可以修改其中之一，重新编译就可以得到自己需要的图片样式效果。

## 3-25图标（一）

**200个icon**：

这里说的图标就是Web制作中常看到的小icon图标，可以说这些小icon图标是一个优秀Web中不可缺少的一部分，起到画龙点睛的效果。在Bootstrap框架中也为大家提供了近**200**个不同的icon图片，而这些图标都是使用**CSS3**的@font-face属性配合字体来实现的icon效果。

[](http://img.mukewang.com/53db0e5b0001aff810560855.jpg)

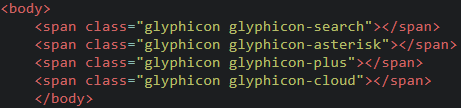
**放心使用**：

在具体介绍Bootstrap的icon图标之前，我们首先要感谢**glyphicons.com**网站，因为Bootstrap框架中图标都是glyphicons.com这个商业网站提供的，并且免费授权给Bootstrap框架使用，所以大家可以放心使用在自己的项目当中。

**原理分析**：

Bootstrap框架中的图标都是字体图标，其实现原理就是通过**@font-face**属性加载了字体。

大家或许会问，这些字体我去哪里获取。如果你是从前面一直阅读过来，我们在介绍文件结构那一节就已介绍过。在Bootstrap框架中有一个**fonts**的目录，这个目录中提供的字体文件就是用于制作icon的字体文件。  
自定义完字体之后，需要对**icon**设置一个默认样式，在Bootstrap框架中是通过给元素添加**“glyphicon”**类名来实现，然后通过伪元素“**:before**”的“**content**”属性调取对应的icon编码：





### 第4章 网格系统

本章介绍Bootstrap框架的网格系统，通过本章的学习，可以使用网格系统完美的实现Web的布局。

Bootstrap 提供了一套响应式、移动设备优先的流式栅格系统，随着屏幕或视口（viewport）尺寸的增加，系统会自动分为最多12列。它包含了易于使用的预定义类，还有强大的mixin 用于生成更具语义的布局。（详见bootstrap官网）

## 实现原理

网格系统的实现原理非常简单，仅仅是通过定义容器大小，平分12份(也有平分成24份或32份，但12份是最常见的)，再调整内外边距，最后结合媒体查询，就制作出了强大的响应式网格系统。Bootstrap框架中的网格系统就是将容器平分成12份。

在使用的时候大家可以根据实际情况重新编译LESS（或Sass）源码来修改12这个数值（也就是换成24或32，当然你也可以分成更多，但不建议这样使用）。

## 4-2工作原理

Bootstrap框架的网格系统工作原理如下：

1、数据行(.row)必须包含在容器（.container）中，以便为其赋予合适的对齐方式和内距(padding)。如：

<div class="container">

<div class="row"></div>

</div>

2、在行(.row)中可以添加列(.column)，但列数之和不能超过平分的总列数，比如12。如：

<div class="container">

<div class="row">

<div class="col-md-4"></div>

<div class="col-md-8"></div>

3、具体内容应当放置在列容器（column）之内，而且只有列（column）才可以作为行容器(.row)的直接子元素

4、通过设置内距（padding）从而创建列与列之间的间距。然后通过为第一列和最后一列设置负值的外距（margin）来抵消内距(padding)的影响

为了更好的理解Bootstrap框架的网格系统工作原理，我们来看一张草图：

[](http://img.mukewang.com/53b0f9c000018b9305540282.jpg)

简单对图解释一下：

1、最外边框，带有一大片白色区域，就是相当于浏览器的可视区域。在Bootstrap框架的网格系统中带有响应式效果，其带有四种类型的浏览器（超小屏，小屏，中屏和大屏），其断点（像素的分界点）是768px、992px和1220px。

2、第二个边框(1)相当于容器(.container)。针对不同的浏览器分辨率，其宽度也不一样：自动、750px、970px和1170px。

3、２号横条阐述的是，将容器的行（.row）平分了12等份，也就是列。每个列都有一个“padding-left:15px”(图中粉红色部分)和一个“padding-right:15px”(图中紫色部分)。这样也导致了第一个列的padding-left和最后一列的padding-right占据了总宽度的30px，从而致使页面不美观，当然，如果你需要留有一定的间距，这个做法是不错的

4、３号横条就是行容器(.row),其定义了“margin-left”和”margin-right”值为”-15px”，用来抵消第一个列的左内距和最后一列的右内距。在bootstrap.css的第763行~第767行可以看到：

.row {

margin-right: -15px;

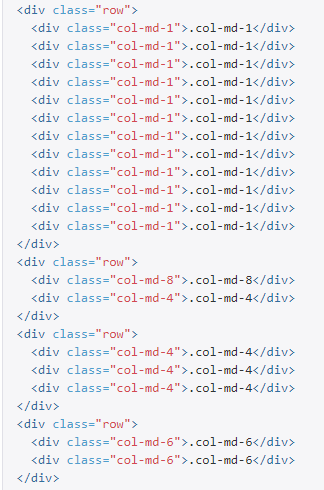
margin-left: -15px;

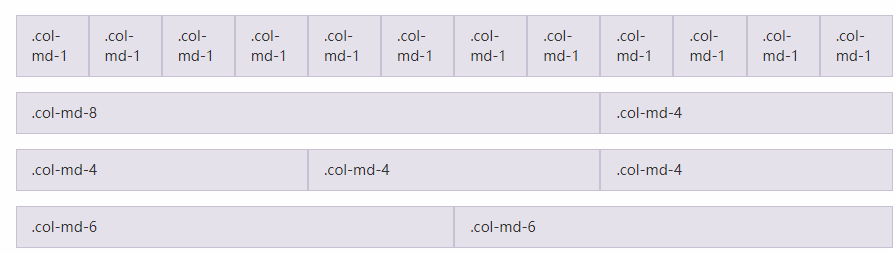
5、将行与列给合在一起就能看到横条4的效果。也就是我们期望看到的效果，第一列和最后一列与容器（.container）之间没有间距。

横条５只是想向大家展示，你可以根据需要，任意组合列与列，只是他们的组合数之和不要超过总列数。

实例：从堆叠到水平排列

使用单一的一组 .col-md-\* 栅格类，就可以创建一个基本的栅格系统，在手机和平板设备上一开始是堆叠在一起的（超小屏幕到小屏幕这一范围），在桌面（中等）屏幕设备上变为水平排列。所有“列（column）必须放在 ” .row 内。





## 4-3基本用法

网格系统用来布局，其实就是列的组合。Bootstrap框架的网格系统中有四种基本的用法。由于Bootstrap框架在不同屏幕尺寸使用了不同的网格样式，在这一节中所涉及到的示例，我们都以中屏（970px）为例进行介绍，其他屏幕的使用也类似这一种。关于屏幕尺寸如下图：



**1、列组合**

列组合简单理解就是更改数字来合并列（原则：列总和数不能超12），有点类似于表格的colspan属性，例如：

<div class="container">

<div class="row">

<div class="col-md-4">.col-md-4</div>

<div class="col-md-8">.col-md-8</div>

</div>

<div class="row">

<div class="col-md-4">.col-md-4</div>

<div class="col-md-4">.col-md-4</div>

<div class="col-md-4">.col-md-4</div>

</div>

<div class="row">

<div class="col-md-3">.col-md-3</div>

<div class="col-md-6">.col-md-6</div>

<div class="col-md-3">.col-md-3</div>

</div>

</div>

使用上面的结构，你将看到下图的效果：



实现列组合方式非常简单，只涉及两个CSS两个特性：**浮动**与**宽度百分比**。在bootstrap.css文件的第1088行～1126行：

/\*确保所有列左浮动\*/

.col-md-1, .col-md-2, .col-md-3, .col-md-4, .col-md-5, .col-md-6, .col-md-7, .col-md-8, .col-md-9, .col-md-10, .col-md-11, .col-md-12 {

float: left;

}

/\*定义每个列组合的宽度（使用的百分比）\*/

.col-md-12 {

width: 100%;

}

.col-md-11 {

width: 91.66666667%;

}

.col-md-10 {

width: 83.33333333%;

}

.col-md-9 {

width: 75%;

}

.col-md-8 {

width: 66.66666667%;

}

.col-md-7 {

width: 58.33333333%;

}

.col-md-6 {

width: 50%;

}

.col-md-5 {

width: 41.66666667%;

}

.col-md-4 {

width: 33.33333333%;

}

.col-md-3 {

width: 25%;

}

.col-md-2 {

width: 16.66666667%;

}

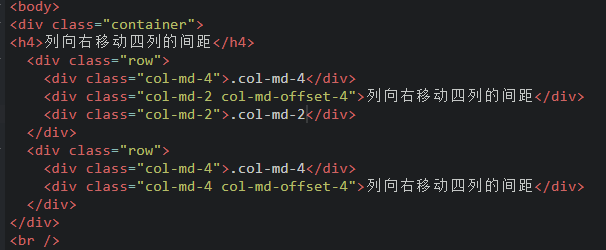
.col-md-1 {

width: 8.33333333%;

}

## 4-4列偏移

有的时候，我们不希望相邻的两个列紧靠在一起，但又不想使用margin或者其他的技术手段来。这个时候就可以使用列偏移（offset）功能来实现。使用列偏移也非常简单，只需要在列元素上添加类名“col-md-offset-\*”(其中星号代表要偏移的列组合数)，那么具有这个类名的列就会向右偏移。例如，你在列元素上添加“col-md-offset-4”，表示该列向右移动4个列的宽度。



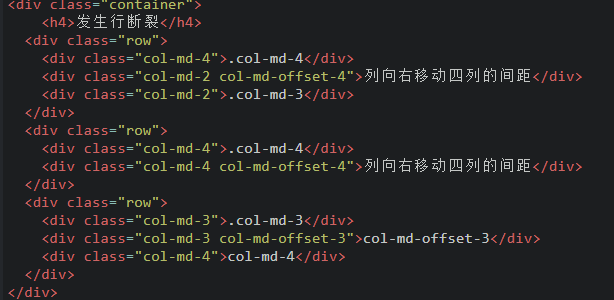


实现原理非常简单，就是利用十二分之一（1/12）的margin-left。然后有多少个offset，就有多少个margin-left

注意：

不过有一个细节需要注意，使用”col-md-offset-\*”对列进行向右偏移时，要保证列与偏移列的总数不超过12，不然会致列断行显示，如：

上面代码中列和偏移列总数为3+3+3+4 = 13>12，所以发生了列断行。





## 4-5列排序

列排序其实就是改变列的方向，就是改变左右浮动，并且设置浮动的距离。在Bootstrap框架的网格系统中是通过添加类名“col-md-push-\*”和“col-md-pull-\*” (其中星号代表移动的列组合数)。

我们来看一个简单的示例：

<div class="container">

<div class="row">

<div class="col-md-4">.col-md-4</div>

<div class="col-md-8">.col-md-8</div>

</div>

</div>

默认情况之下，上面的代码效果如下：

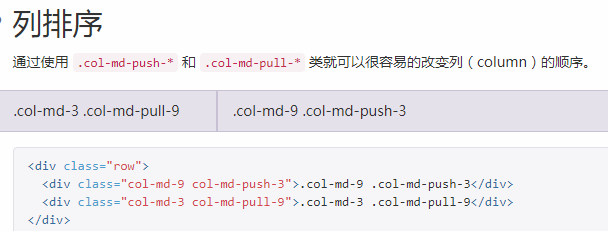
[http://img.mukewang.com/53b10a400001af8005540037.jpg](http://img.mukewang.com/53b10a400001af8005540037.jpg)

“col-md-4”居左，“col-md-8”居右，如果要互换位置，需要将“col-md-4”向右移动８个列的距离，也就是8个offset ,也就是在“<div class=“col-md-4”>”添加类名“col-md-push-8”，调用其样式。

[http://img.mukewang.com/53b10a79000152b805540042.jpg](http://img.mukewang.com/53b10a79000152b805540042.jpg)

也要将“col-md-8”向左移动４个列的距离，也就是4个offset，在“<div class=”col-md-8”>”上添加类名“col-md-pull-4”：

[http://img.mukewang.com/53b10aa80001687005530038.jpg](http://img.mukewang.com/53b10aa80001687005530038.jpg)



## 4-6列的嵌套

Bootstrap框架的网格系统还支持列的嵌套。你可以在一个列中添加一个或者多个行（row）容器，然后在这个行容器中插入列（像前面介绍的一样使用列）。但在列容器中的行容器（row），宽度为100%时，就是当前外部列的宽度。来看一个简单示例：





注意：嵌套的列总数也需要遵循不超过12列。不然会造成末位列换行显示。

### 第5章 菜单、按钮及导航

菜单（导航菜单）是一个Web网站或者Web应用程序必不可少的一个组件。在Bootstrap框架中也提供了这样的组件。在这一节中，我们主要和大家探讨下拉菜单组件的使用。

## 5-1下拉菜单（基本用法）

小伙伴们注意，在Bootstrap框架中的**下拉菜单组件**是一个独立的组件，根据不同的版本，它对应的文件：

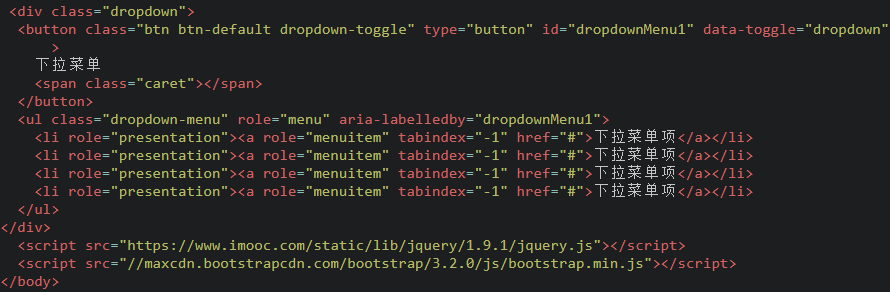
  ☑  LESS版本：对应的源码文件为 **dropdowns.less**

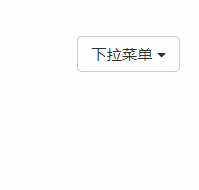
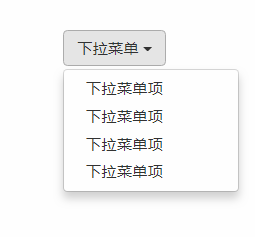
  ☑  Sass版本：对应的源码文件为 **\_dropdowns.scss**

  ☑  编译后的Bootstrap版本：查看bootstrap.css文件第3004行～第3130行

在使用Bootstrap框架的下拉菜单时，必须调用Bootstrap框架提供的bootstrap.js文件。当然，如果你使用的是未编译版本，在js文件夹下你能找到一个名为“dropdown.js”的文件。你也可以调用这个js文件。不过在我们的教程中，我们统一调用压缩好的“bootstrap.min.js”文件：

特别声明：因为Bootstrap的组件交互效果都是依赖于jQuery库写的插件，所以在使用bootstrap.min.js之前一定要先加载jquery.min.js才会生效果



**使用方法：**

在使用Bootstrap框架中的下拉菜单组件时，其结构运用的正确与否非常的重要，如果结构和类名未使用正确，直接影响组件是否能正常运用。我们来简单的看看：

1、使用一个名为“**dropdown**”的容器包裹了整个下拉菜单元素，示例中为:

<div class="dropdown"></div>

2、使用了一个**<button>**按钮做为父菜单，并且定义类名“**dropdown-toggle**”和自定义“**data-toggle**”属性，且值必须和最外容器类名一致，此示例为:

**data-toggle="dropdown"**

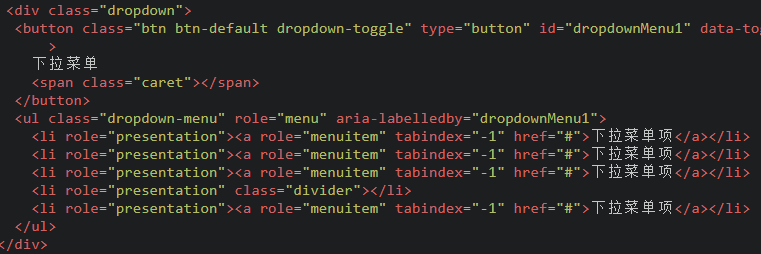
3、下拉菜单项使用一个**ul**列表，并且定义一个类名为“**dropdown-menu**”，此示例为:

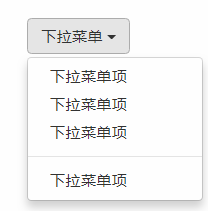
<ul class="**dropdown-menu**">

bootstrap框架中的下拉菜单组件，其下拉菜单项默认是隐藏的, 因为“dropdown-menu”默认样式设置了“display:none”

## 5-3下拉菜单（下拉分隔线）

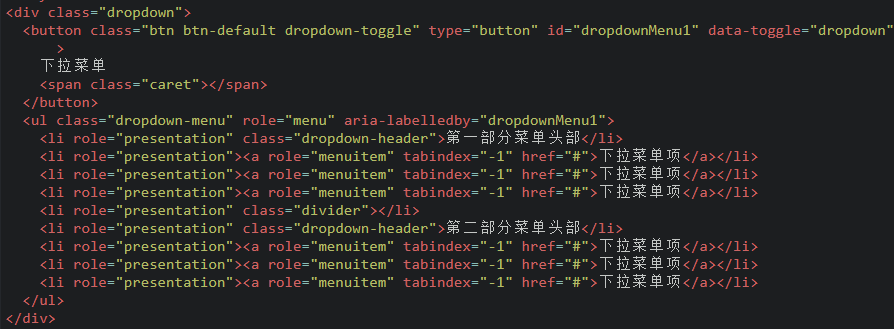
在Bootstrap框架中的下拉菜单还提供了下拉分隔线，假设下拉菜单有两个组，那么组与组之间可以通过添加一个空的<li>，并且给这个<li>添加类名“divider”来实现添加下拉分隔线的功能。





## 5-4下拉菜单（菜单标题）

上一小节讲解通过添加“divider”可以将下拉菜单分组，为了让这个分组更明显，还可以给每个组添加一个头部（标题）, <li role="presentation" **class="dropdown-header"**>第一部分菜单头部</li>





## 5-5下拉菜单（对齐方式）

**实现右对齐方法：**

Bootstrap框架中下拉菜单默认是左对齐，如果你想让下拉菜单相对于父容器**右对齐**时，可以在“dropdown-menu”上添加一个“pull-right”或者“dropdown-menu-right”类名，如下所示：

<div class="dropdown">

<button class="btn btn-default dropdown-toggle" type="button" id="dropdownMenu1" data-toggle="dropdown">

下拉菜单

<span class="caret"></span>

</button>

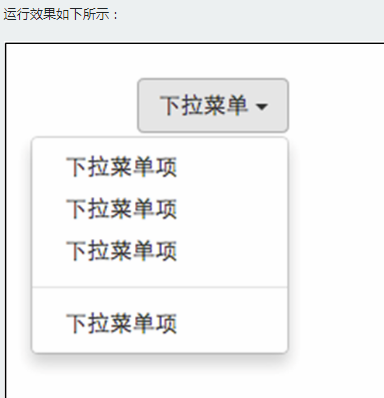
<ul class="dropdown-menu **pull-right**" role="menu" aria-labelledby="dropdownMenu1">

…

</ul>

</div>

上面代码中的“pull-right”类可以用“dropdown-menu-right”代替，两个类的作用是一样的



## 5-6下拉菜单（菜单项状态）

下拉菜单项的默认的状态（不用设置）有悬浮状态（:hover）和焦点状态（:focus）：

下拉菜单项除了上面两种状态，还有当前状态（.active）和禁用状态（.disabled）。这两种状态使用方法只需要在对应的菜单项上添加对应的类名：



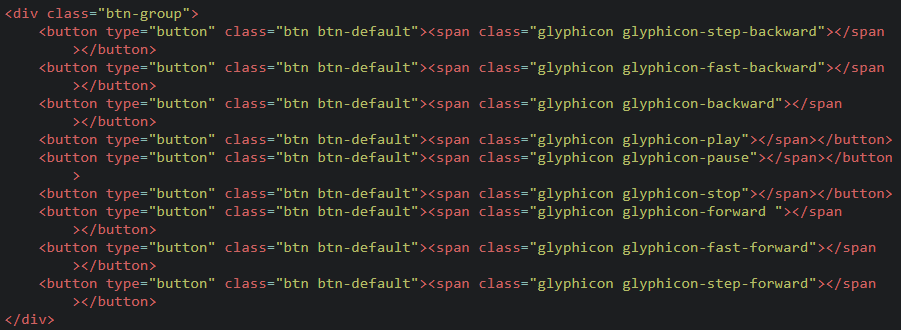
## 5-7按钮（按钮组）

单个按钮在Web页面中的运用有时候并不能满足我们的业务需求，常常会看到将多个按钮组合在一起使用，比如富文本编辑器里的一组小图标按钮等。那么在这一节中，我们主要向大家介绍Bootstrap框架为大家提供的按钮组组件。

**使用方法：**

按钮组和下拉菜单组件一样，需要依赖于button.js插件才能正常运行。不过我们同样可以直接只调用bootstrap.js文件。因为这个文件已集成了button.js插件功能。

对于结构方面，非常的简单。使用一个名为“btn-group”的容器，把多个按钮放到这个容器中





从效果图上我们可以看出，按钮组四个角都是圆角（支持CSS3的浏览器），但有的小伙伴会问，我们平常制作网页时每个按钮都是带有圆角，而在按钮组中的按钮，除了第一个和最后一个具有边上的圆角之外，其他的按钮没有圆角，它是怎么实现的呢？其实实现方法非常简单：

  1、默认所有按钮都有圆角

  2、除第一个按钮和最后一个按钮（下拉按钮除外），其他的按钮都取消圆角效果

  3、第一个按钮只留左上角和左下角是圆角

  4、最后一个按钮只留右上角和右下角是圆角

## 5-8按钮（按钮工具栏）

在富文本编辑器中，将按钮组分组排列在一起，比如说复制、剪切和粘贴一组；左对齐、中间对齐、右对齐和两端对齐一组，如下图所示：

[](http://img.mukewang.com/53e45edc00019ad308600101.jpg)

那么Bootstrap框架按钮工具栏也提供了这样的制作方法，你只需要将按钮组“btn-group”按组放在一个大的容器“btn-toolbar”中，如下所示：

<div class="btn-toolbar">

<div class="btn-group">

…

</div>

<div class="btn-group">

…

</div>

<div class="btn-group">

…

</div>

<div class="btn-group">

…

</div>

</div>

实现原理主要是让容器的多个分组“btn-group”元素进行浮动，并且组与组之前保持5px的左外距。



运行效果如下：



**按钮组大小设置**

在介绍按钮一节中，我们知道按钮是通过btn-lg、btn-sm和btn-xs三个类名来调整padding、font-size、line-height和border-radius属性值来改变按钮大小。那么按钮组的大小，我们也可以通过类似的方法：

  ☑  .btn-group-lg:大按钮组

  ☑  .btn-group-sm:小按钮组

  ☑  .btn-group-xs:超小按钮组

只需要在“.btn-group”类名上追加对应的类名，就可以得到不同大小的按钮组。如下所示：

<div class="btn-toolbar">

<div class="btn-group **btn-group-lg**">

…

</div>

<div class="btn-group">

…

</div>

<div class="btn-group **btn-group-sm**">

…

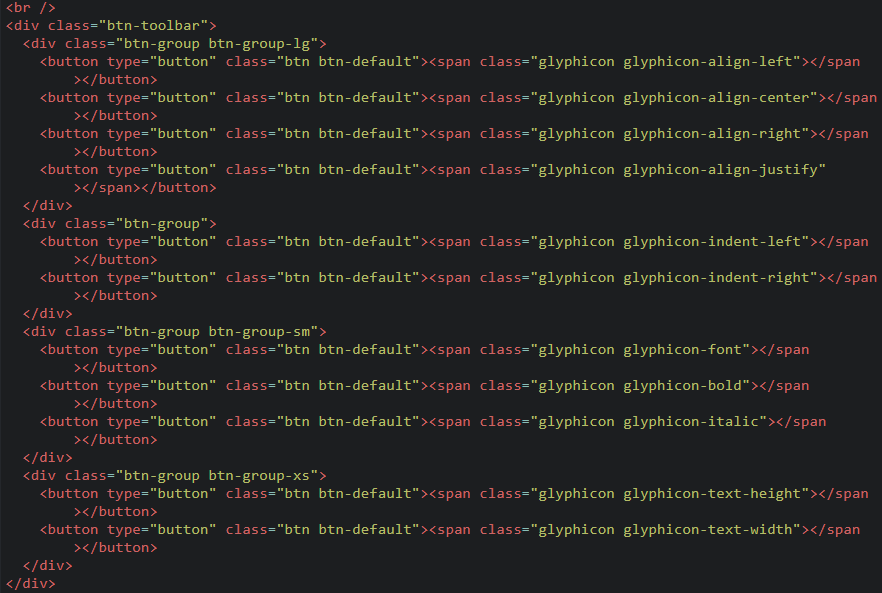
</div>

<div class="btn-group **btn-group-xs**">

…

</div>

</div>



运行效果如下所示：

[](http://img.mukewang.com/53e4632b0001bb2808230100.jpg)

## 5-9按钮（嵌套分组）

很多时候，我们常把下拉菜单和普通的按钮组排列在一起，实现类似于导航菜单的效果。如下所示：

[](http://img.mukewang.com/53e466ac0001273008410307.jpg)

使用的时候，只需要把当初制作下拉菜单的“dropdown”的容器换成“btn-group”，并且和普通的按钮放在同一级





## 5-10按钮（垂直分组）

前面看到的示例，按钮组都是水平显示的。但在实际运用当中，总会碰到垂直显示的效果。在Bootstrap框架中也提供了这样的风格。我们只需要把水平分组的“btn-group”类名换成“btn-group-vertical”即可





## 按钮（等分按钮）

等分按钮的效果在移动端上特别的实用。整个按钮组宽度是容器的100%，而按钮组里面的每个按钮平分整个容器宽度。例如，如果你按钮组里面有五个按钮，那么每个按钮是20%的宽度，如果有四个按钮，那么每个按钮是25%宽度，以此类推。

等分按钮也常被称为是自适应分组按钮，其实现方法也非常的简单，只需要在按钮组“btn-group”上追加一个“**btn-group-justified**”类名，如下所示：

<div class="btn-wrap">

<div class="btn-group **btn-group-justified**">

<a class="btnbtn-default" href="#">首页</a>

<a class="btnbtn-default" href="#">产品展示</a>

<a class="btnbtn-default" href="#">案例分析</a>

<a class="btnbtn-default" href="#">联系我们</a>

</div>

</div>

运行效果如下：

[](http://img.mukewang.com/53e46af60001ab0306850099.jpg)

实现原理非常简单，把“btn-group-justified”模拟成表格（display:table），而且把里面的按钮模拟成表格单元格（display:table-cell）。

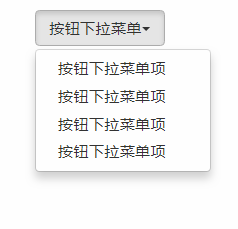
**特别声明**：在制作等分按钮组时，请尽量使用**<a>**标签元素来制作按钮，因为使用<button>标签元素时，使用display:table在部分浏览器下支持并不友好。

## 5-12按钮下拉菜单

按钮下拉菜单仅从外观上看和上一节介绍的下拉菜单效果基本上是一样的。不同的是在普通的下拉菜单的基础上封装了按钮（.btn）样式效果。简单点说就是点击一个按钮，会显示隐藏的下拉菜单。

按钮下拉菜单其实就是普通的下拉菜单，只不过把“<a>”标签元素换成了“<button>”标签元素。唯一不同的是外部容器“div.dropdown”换成了“div.btn-group”。





## 5-13按钮的向下向上三角形

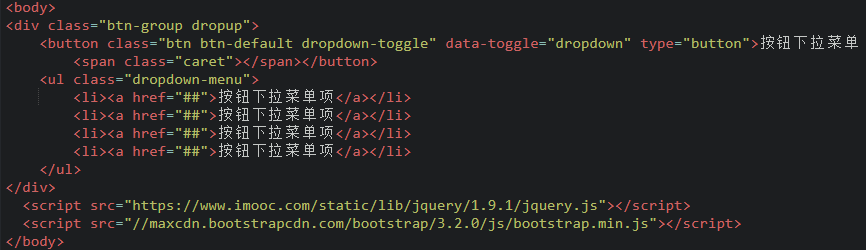
按钮的向下三角形，我们是通过在<button>标签中添加一个“<span>”标签元素，并且命名为“caret”:

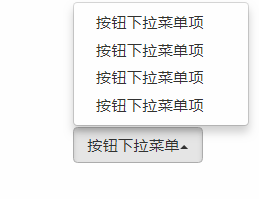
<button class="btn btn-default **dropdown-toggle**" **data-toggle="dropdown"** type="button">按钮下拉菜单<span class="caret"></span></button>

这个三角形完全是通过CSS代码来实现的：

另外在按钮中的三角形“caret”做了一定的样式处理：

有的时候我们的下拉菜单会向上弹起（接下来一个小节会介绍），这个时候我们的三角方向需要朝上显示，实现方法：需要在“.btn-group”类上追加“dropup”类名（这也是做向上弹起下拉菜单要用的类名）。



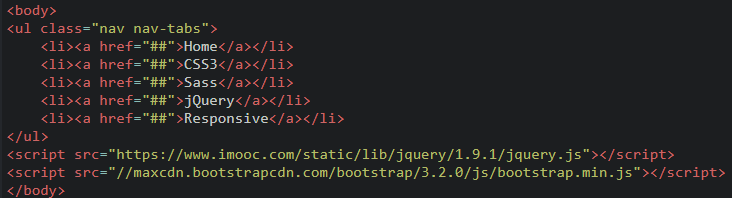


## 5-15导航（基础样式）

导航对于一位前端人员来说并不陌生。可以说导航是一个网站重要的元素组件之一，可以便于用户查找网站所提供的各项功能服务。导航的制作方法也是千奇百怪，五花八门。在这一节中将向大家介绍如何使用Bootstrap框架制作各式各样的导航。

**导航基础样式**

Bootstrap框架中制作导航条主要通过“.nav”样式。默认的“.nav”样式不提供默认的导航样式，必须附加另外一个样式才会有效，比如“nav-tabs”、“nav-pills”之类。比如右侧代码编辑器中就有一个tab导航条的例子，他的实现方法就是为ul标签加入.nav和nav-tabs两个类样式。





## 5-16导航（标签形tab导航）

标签形导航，也称为选项卡导航。特别是在很多内容分块显示的时，使用这种选项卡来分组十分适合。

标签形导航是通过“**nav-tabs**”样式来实现。在制作标签形导航时需要在原导航“nav”上追加此类名，如：

<ul class="**nav nav-tabs**">

<li><a href="##">Home</a></li>

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

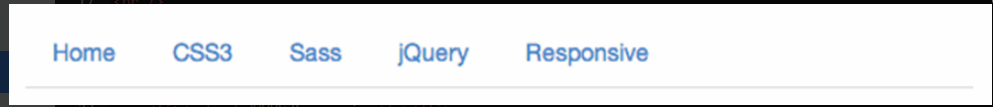
<li><a href="##">Sass</a></li>

<li><a href="##">jQuery</a></li>

<li><a href="##">Responsive</a></li>

</ul>

运行的效果如下所示：



实现原理非常的简单，将菜单项（li）按块显示，并且让他们在同一水平上排列，然后定义非高亮菜单的样式和鼠标悬浮效果。

其实上例的效果和我们平时看到的选项卡效果并不一致。一般情况之下，选项卡教会有一个当前选中项。其实在Bootstrap框架也相应提供了。假设我们想让“Home”项为当前选中项，只需要在其标签上添加类名“class="active"”即可：

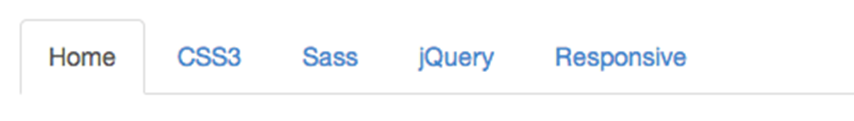
<ul class="nav nav-tabs">

<li **class="active"**><a href="##">Home</a></li>

…

</ul>

运行效果如下：

[](http://img.mukewang.com/53e86b7700019e0308540121.jpg)

除了当前项之外，有的选项卡还带有禁用状态，实现这样的效果，只需要在标签项上添加“class="disabled"”即可：

<ul class="nav nav-tabs">

<li class="active"><a href="##">Home</a></li>

…

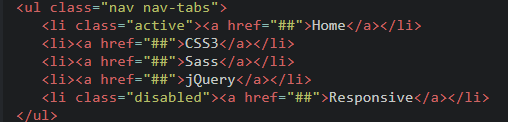
<li **class="disabled"**><a href="##">Responsive</a></li>

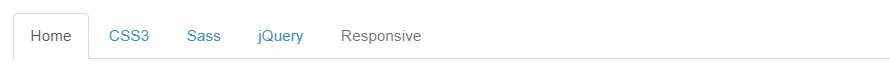
</ul>

运行效果如下：

[](http://img.mukewang.com/53e86c8b00015ca208550155.jpg)

实现这个效果的样式，在默认样式“.nav”中就带有：





注意：我们看到的选项卡效果，点击菜单项就可以切换内容，如果要实现这样的效果需要配合js插件，这部分将在后面的教程中会介绍。

## 5-17导航（胶囊形(pills)导航）

胶囊形（pills）导航听起来有点别扭，因为其外形看起来有点像胶囊形状。但其更像我们平时看到的大众形导航。当前项高亮显示，并带有圆角效果。其实现方法和“nav-tabs”类似，同样的结构，只需要把类名“nav-tabs”换成“**nav-pills**”即可：

<ul class="**nav nav-pills**">

<li class="**active**"><a href="##">Home</a></li>

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

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

<li><a href="##">jQuery</a></li>

<li class="**disabled**"><a href="##">Responsive</a></li>

</ul>

[](http://img.mukewang.com/53e86ee60001711e08160307.jpg)

## 5-18导航（垂直堆叠的导航）

在实际运用当中，除了水平导航之外，还有垂直导航，就类似前面介绍的垂直排列按钮一样。制作垂直堆叠导航只需要在“nav-pills”的基础上添加一个“nav-stacked”类名即可：

<ul class="nav nav-pills **nav-stacked**">

<li class="active"><a href="##">Home</a></li>

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

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

<li><a href="##">jQuery</a></li>

<li class="disabled"><a href="##">Responsive</a></li>

</ul>

运行效果如下：

[](http://img.mukewang.com/53e871a2000102b707240444.jpg)

大家是否还记得，在下拉菜单一节中，下拉菜单组与组之间有一个分隔线。其实在垂直堆叠导航也具有这样的效果，只需要添加在导航项之间添加“<li class=”nav-divider”></li>”即可：

<ul class="nav nav-pills nav-stacked">

<li class="active"><a href="##">Home</a></li>

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

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

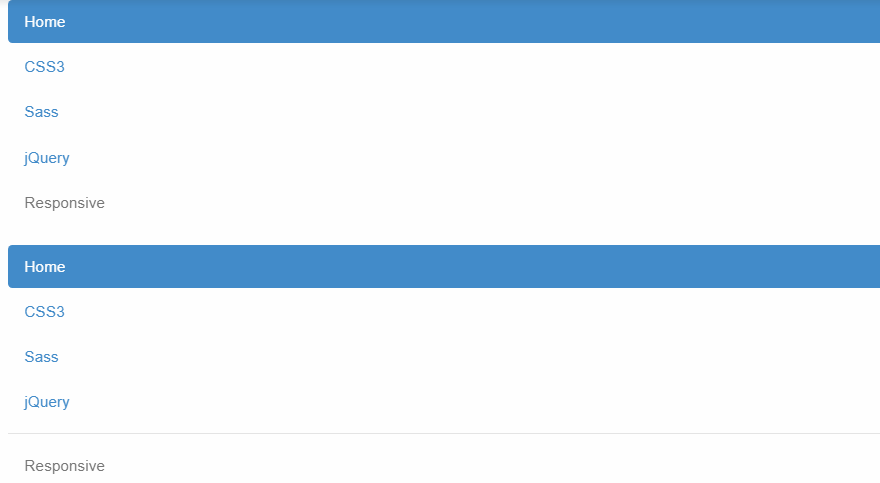
<li><a href="##">jQuery</a></li>

**<li class="nav-divider"></li>**

<li class="disabled"><a href="##">Responsive</a></li>

</ul>

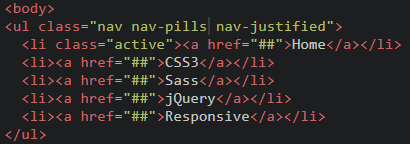
垂直堆叠导航与胶囊形导航相比，主要是让导航项不浮动，让其垂直排列，然后给相邻导航项留有一定的间距：



或许你会问，如果我在”nav-tabs”上添加“nav-stacked”是不是也能实现垂直的标签选项导航呢？答案是：在bootstrap V2.x版本可以，但在Bootstrap V3.x版本将这个效果取消了，可能作者觉得垂直选择项并不太常见，也不美观吧。

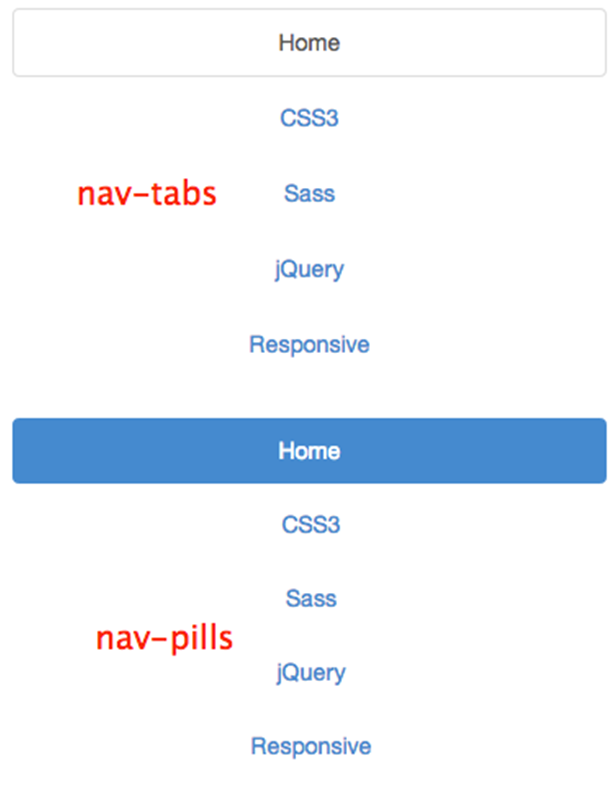
## 5-19自适应导航（使用）

自适应导航指的是导航占据容器全部宽度，而且菜单项可以像表格的单元格一样自适应宽度。自适应导航和前面使用“btn-group-justified”制作的自适应按钮组是一样的。只不过在制作自适应导航时更换了另一个类名“**nav-justified**”。当然他需要和“nav-tabs”或者“nav-pills”配合在一起使用。如：





实现原理并不难，列表（<ul>）上设置宽度为“100%”，然后每个菜单项(<li>)设置了“display:table-cell”，让列表项以模拟表格单元格的形式显示，这里有一个媒体查询条件：“@media (min-width:768px){…}”表示自适应导航仅在浏览器视窗宽度大于768px才能按上图风格显示。当你的浏览器视窗宽度小于768px的时候，将会按下图的风格展示：

[](http://img.mukewang.com/53e874f70001bacb06150786.jpg)

## 5-21导航加下拉菜单（二级导航）

前面介绍的都是使用Bootstrap框架制作一级导航，但很多时候，在Web页面中是离不开二级导航的效果。那么在Bootstrap框架中制作二级导航就更容易了。只需要将li当作父容器，使用类名“dropdown”，同时在li中嵌套另一个列表ul，使用前面介绍下拉菜单的方法就可以：

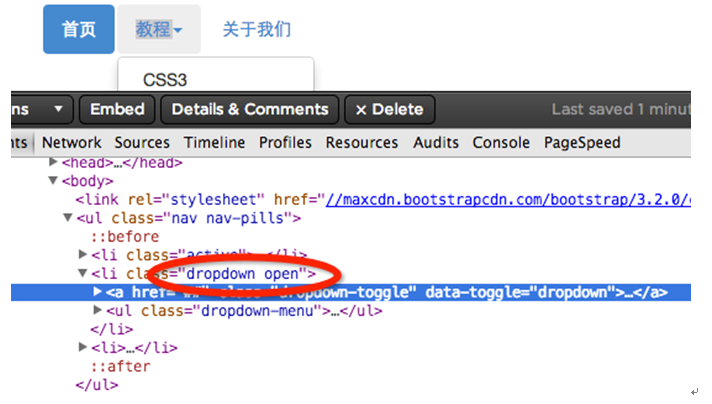






Bootstrap很多组件和插件都是通过这样的data属性来完成

通过浏览器调试工具，不难发现，点击有二级导航的菜单项，会自动添加“open”类名，再次点击就会删除添加的“open”类名：

[](http://img.mukewang.com/53e878580001e0b307120404.jpg)

简单点来说，就是依靠这个类名来控制二级导航显示与否，并且设置了背景色和边框：

大家回忆一下，在制作下拉菜单时，可以用分隔线，那么在二级导航中是否可以呢？我们一起来看看：

不用再说太多，只需要添加“<li class=”nav-divider”></li>”这样的一个空标签就可以了。

运行效果如下：

## 5-22面包屑式导航

面包屑(Breadcrumb)一般用于导航，主要是起的作用是告诉用户现在所处页面的位置（当前位置）。在Bootstrap框架中面包屑也是一个独立模块组件：

使用方式就很简单，为ol加入breadcrumb类：

<ol class="**breadcrumb**">

<li><a href="#">首页</a></li>

<li><a href="#">我的书</a></li>

<li class="**active**">《图解CSS3》</li>

</ol>



### 第6章 导航条、分页导航

导航条和上一章介绍的导航，就相差一个字，多了一个“条”字。其实在Bootstrap框架中他们还是明显的区别。在导航条中有一个背景色、而且导航条可以是纯链接（类似导航），也可以是表单，还有就是表单和导航一起结合等多种形式。

## 6-1导航条基础

导航条（navbar）和上一节介绍的导航（nav），就相差一个字，多了一个“条”字。其实在Bootstrap框架中他们还是明显的区别。在导航条(navbar)中有一个背景色、而且导航条可以是纯链接（类似导航），也可以是表单，还有就是表单和导航一起结合等多种形式。在这一节中将一起探讨Bootstrap框架中导航条的使用。

导航条和导航一样，在Bootstrap框架中是一个独立组件，所以你也可以根据自己的需求使用不同的版本：

* **LESS版本**：对应的源文件navbar.less
* **Sass版本**：对应的源文件\_navbar.scss
* **编译后的版本**：查看bootstrap.css文件第3642行～第4111行（注意这个所说的Bootstrap版本是3.1.1，其它版本位置是不一致的）

## 6-2基础导航条

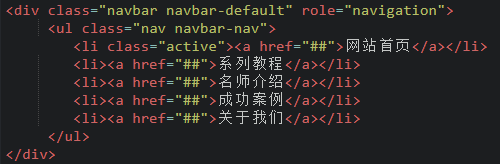
在Bootstrap框中，导航条和导航从外观上差别不是太多，但在实际使用中导航条要比导航复杂得多。我们先来看导航条中最基础的一个——基础导航条。

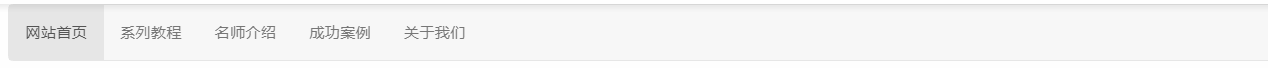
**使用方法：**

在制作一个基础导航条时，主要分以下几步：

**第一步：**首先在制作导航的列表(<ul class=”nav”>)基础上添加类名“navbar-nav”

**第二步：**在列表外部添加一个容器（div），并且使用类名“navbar”和“navbar-default”





“.navbar”样式的主要功能就是设置左右padding和圆角等效果，但他和颜色相关的样式没有进行任何的设置，而导航条的颜色都是通过“.navbar-default”来进行控制

navbar-nav样式是在导航.nav的基础上重新调整了菜单项的浮动与内外边距。同时也不包括颜色等样式设置，而颜色和其他样式是通过配合父容器“navbar-default”来一起实现

## 6-3为导航条添加标题、二级菜单及状态

**加入导航条标题**

在Web页面制作中，常常在菜单前面都会有一个标题（文字字号比其它文字稍大一些），其实在Bootstrap框架也为大家做了这方面考虑，其通过“navbar-header”和“navbar-brand”来实现





此功能主要起一个提醒功能，当然改良一下可以当作是logo(此处不做过多阐述)。其样式主要是加大了字体设置，并且设置了最大宽度。

**导航条状态、二级菜单**

同样的，在基础导航条中对菜单提供了当前状态，禁用状态，悬浮状态等效果，而且也可以带有二级菜单的导航条，案例代码见右侧代码编辑器（24-42行）。

效果图如下：

[](http://img.mukewang.com/53f55cad00018e8008660190.jpg)



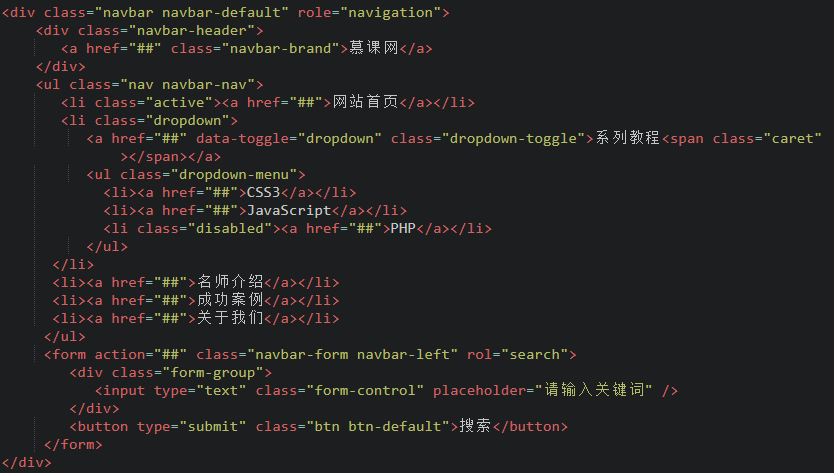


## 6-4带表单的导航条

有的导航条中会带有搜索表单，比如新浪微博的导航条：

[http://img.mukewang.com/53edcf9d00013cf506950056.jpg](http://img.mukewang.com/53edcf9d00013cf506950056.jpg)

在Bootstrap框架中提供了一个“navbar-form”，使用方法很简单，在navbar容器中放置一个带有navbar-form类名的表单，示例代码编辑器（29-34行）。





在上面的示例中，大家看到了“navbar-left”让表单左浮动，更好实现对齐。在Bootstrap框架中，还提供了“navbar-right”样式，让元素在导航条靠右对齐。

注意：这里有一个条件，只有当浏览器视窗宽度大于768px生效。

**6-5导航条中的按钮、文本和链接**

Bootstrap框架的导航条中除了使用navbar-brand中的a元素和navbar-nav的ul和navbar-form之外，还可以使用其他元素。框架提供了三种其他样式：

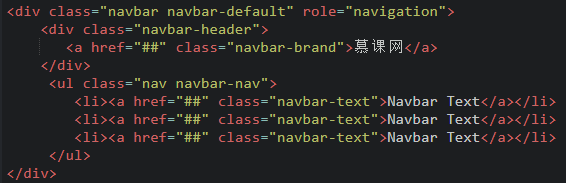
1、导航条中的按钮**navbar-btn**

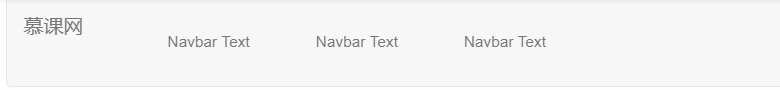
2、导航条中的文本**navbar-text**

3、导航条中的普通链接**navbar-link**

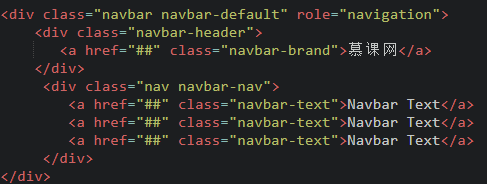
但这三种样式在框架中使用时受到一定的限制，需要和navbar-brand、navbar-nav配合起来使用。而且对数量也有一定的限制，一般情况在使用一到两个不会有问题，超过两个就会有问题。

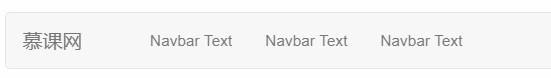
**navbar-btn和navbar-link在上一小节已经讲过，现在我们**来看一下**navbar-text**的一个简单应用（见右侧代码编辑区11-20行）





在结果窗口中查看效果，可以看出明显的存在问题，通过浏览器调试工具，可以看到a标签有margin-top和margin-bottom的值为15px，欲将其对齐，我们将上面的结构做一定的调整







## 6-6固定导航条

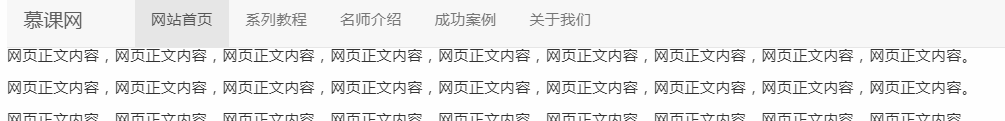
很多情况之一，设计师希望导航条固定在浏览器顶部或底部，这种固定式导航条的应用在移动端开发中更为常见。Bootstrap框架提供了两种固定导航条的方式：

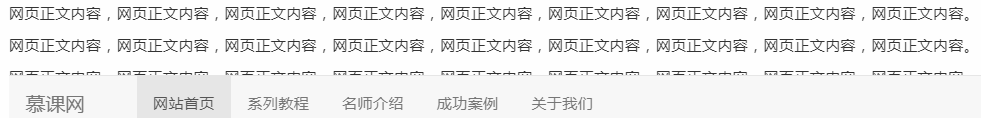
   ☑  **.navbar-fixed-top**：导航条固定在浏览器窗口顶部

   ☑  **.navbar-fixed-bottom**：导航条固定在浏览器窗口底部

使用方法很简单，只需要在制作导航条最外部容器navbar上追加对应的类名即可







实现原理很简单，就是在navbar-fixed-top和navbar-fixed-bottom使用了position：fixed属性，并且设置navbar-fixed-top的top值为0,而navbar-fixed-bottom的bottom值为0

**存在bug及解决方法:（亲测可行）**

从运行效果中大家不难发现，页面主内容顶部和底部都被固定导航条给遮住了。为了避免固定导航条遮盖内容，我们需要在body上做一些处理：

body {

padding-top: 70px;/\*有顶部固定导航条时设置\*/

padding-bottom: 70px;/\*有底部固定导航条时设置\*/

}



因为固定导航条默认高度是50px，我们一般设置padding-top和padding-bottom的值为70px，当然有的时候还是需要具体情况具体分析。

**第二种解决这个bug方法：**

其实除了这种解决方案之外，我们还有其他的解决方法，把固定导航条都放在页面内容前面：

<div class="navbar navbar-default navbar-fixed-top" role="navigation">

　…

</div>

<div class="navbar navbar-default navbar-fixed-bottom" role="navigation">

　…

</div>

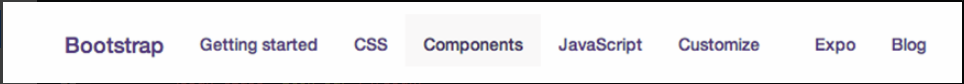
<div class="content">我是内容</div>

## 6-7响应式导航条

如今浏览Web页面的终端不在是一尘不变了，前面示例实现的导航条仅能适配于大屏幕的浏览器，但当浏览器屏幕变小的时候，就不适合了。因此响应式设计也就随之而来。那么在一个响应式的Web页面中，对于响应式的导航条也就非常的重要。例如Bootstrap框架官网的导航条：

[http://img.mukewang.com/53eded3b0001db2a06970046.jpg](http://img.mukewang.com/53eded3b0001db2a06970046.jpg)

（宽屏时效果）



（中屏时效果）

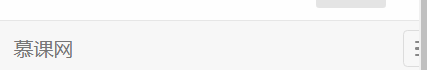
[](http://img.mukewang.com/53f580e30001bba208690079.jpg)

（窄屏时效果）

那么在这一节中，主要和大家探讨如何使用Bootstrap框架实现响应式导航条效果。

先来看HTML结构见右侧代码区。







**使用方法：**

1. 保证在窄屏时需要折叠的内容必须包裹在带一个div内，并且为这个div加入collapse、navbar-collapse两个类名。最后为这个div添加一个class类名或者id名。



2、保证在窄屏时要显示的图标样式（固定写法）：

<button class="navbar-toggle" type="button" data-toggle="collapse">

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

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

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

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

</button>

3、并为button添加data-target=".类名/#id名"，究竞是类名还是id名呢？由需要折叠的div来决定。如：

需要折叠的div代码段：

<div class="collapse navbar-collapse" id="example">

<ul class="nav navbar-nav">

…

</ul>

</div>

窄屏时显示的图标代码段：

<button class="navbar-toggle" type="button" data-toggle="collapse" data-target="#example">

...

</button>

也可以这么写，需要折叠的div代码段：

<div class="collapse navbar-collapse example" >

<ul class="nav navbar-nav">

…

</ul>

</div>

窄屏时要显示的图标：

<button class="navbar-toggle" type="button" data-toggle="collapse" data-target=".example">

...

</button>

## 6-8反色导航条

反色导航条其实是Bootstrap框架为大家提供的第二种风格的导航条，与默认的导航条相比，使用方法并无区别，只是将navbar-deafult类名换成navbar-inverse。其变化只是导航条的背景色和文本做了修改





## 6-9分页导航（带页码的分页导航）

分页导航几乎在哪个网站都可见。好的分页导航能给用户带来更好的用户体验。在Bootstrap框架中提供了两种分页导航：

   ☑   带页码的分页导航

   ☑   带翻页的分页导航

**带页码的分页导航**

带页码的分页导航，可能是最常见的一种分页导航，特别是在列表页内容超多的时候，会给用户提供分页的导航方式。在Bootstrap框架为开发者提供不同的版本：

   ☑   LESS版本：对应的源文件pagination.less

   ☑   Sass版本：对应的源文件\_pagination.scss

   ☑   编译后版本：对应bootstrap.css文件第4130行～第4222行

**使用方法：**

平时很多同学喜欢用div>a和div>span结构来制作带页码的分页导航。不过，在Bootstrap框架中使用的是ul>li>a这样的结构，在ul标签上加入pagination方法：

<ul class="pagination">

<li><a href="#">&laquo;</a></li>

<li><a href="#">1</a></li>

<li><a href="#">2</a></li>

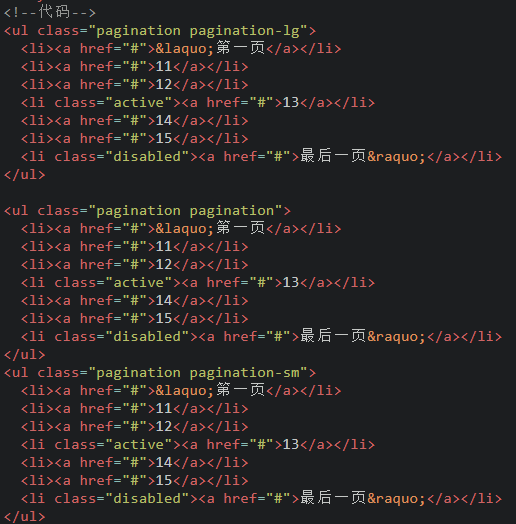
<li><a href="#">3</a></li>

<li><a href="#">4</a></li>

<li><a href="#">5</a></li>

<li><a href="#">&raquo;</a></li>

</ul>





**实现原理：**

从效果中可以看出，当前状态页码会高亮显示，而且不能点击。而最后一页是禁用状态，也不能点击

注意：要禁用当前状态和禁用状态不能点击，我们还要依靠js来实现，或者将这两状态下的a标签换成span标签。

**大小设置：**

在Bootstrap框架中，也可以通过几个不同的情况来设置其大小。类似于按钮一样：

1、通过“pagination-lg”让分页导航变大；

2、通过“pagination-sm”让分页导航变小：

**大小设置实现原理：**

其实就是通增加相应的padding大小、font-size大小和圆角大小

## 6-10分页导航（翻页分页导航）

Bootstrap框架除了提供带页码的分页导航之外还提供了翻页导航。这种分页导航常常在一些简单的网站上看到，比如说个人博客，杂志网站等。这种分页导航是看不到具体的页码，只会提供一个“上一页”和“下一页”的按钮。

Bootstrap框架将其独立成一个单独的部分：

  ☑   LESS版本：对应源文件为pager.less

  ☑  Sass版本：对应源文件为\_pager.scss

  ☑  编译后版本：对应bootstrap.css文件第4223行～第4260行

**使用方法：**

在实际使用中，翻页分页导航和带页码的分页导航类似，为ul标签加入pager类：

<ul class="**pager**">

<li><a href="#">&laquo;上一页</a></li>

<li><a href="#">下一页&raquo;</a></li>

</ul>

**对齐样式设置：**

默认情况之下，翻页分页导航是居中显示，但有的时候我们需要一个居左，一个居右。Bootstrap框架提供了两个样式：

   ☑   previous：让“上一步”按钮居左

   ☑   next：让“下一步”按钮居右

具体使用的时候，只需要在li标签上添加对应类名即可：

<ul class="pager">

<li class="**previous**"><a href="#">&laquo;上一页</a></li>

<li class="**next**"><a href="#">下一页&raquo;</a></li>

</ul>

实现原理很简单，就是一个进行了左浮动，一个进行了右浮动

**状态样式设置：**

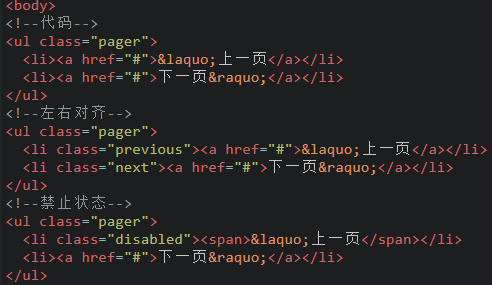
和带页码分页导航一样，如果在li标签上添加了disabled类名的时候，分页按钮处于禁用状态，但同样不能禁止其点击功能。你可以通过js来处理，或将a标签换成span标签。

<ul class="pager">

**<li class="disabled"><span>&laquo;上一页</span></li>**

<li><a href="#">下一页&raquo;</a></li>

</ul>





## 6-11标签

在一些Web页面中常常会添加一个标签用来告诉用户一些额外的信息，比如说在导航上添加了一个新导航项，可能就会加一个“new”标签，来告诉用户。这是新添加的导航项。如下图所示：

[](http://img.mukewang.com/53f5a3810001256d05550068.jpg)

那么在Bootstrap框架中特意将这样的效果提取出来成为一个标签组件，并且以“.label”样式来实现高亮显示。

既然他是一个独立的组件，当然在不同的版本下有不同的文件：

   ☑   LESS版本：对应的源文件label.less

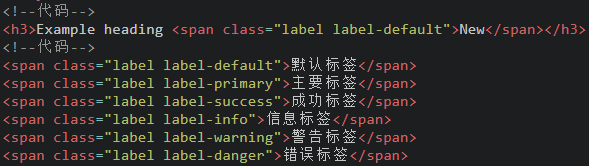
   ☑   Sass版本：对应的源文件\_label.scss

   ☑   编译后版本：bootstrap.css文件第4261行～第4327行

**使用原理：**

使用方法很简单，你可以在使用span这样的行内标签：

<h3>Example heading **<span class="label label-default">New</span>**</h3>





## 6-12徽章

从某种意义上来说，徽章效果和前面介绍的标签效果是极其的相似。也是用来做一些提示信息使用。常出现的是一些系统发出的信息，比如你登录你的twitter后，如果你信息没有看，系统会告诉你有多少信息未读，如下图所示：

[](http://img.mukewang.com/53f5aac500010a7f04370079.jpg)

在Bootstrap框架中，把这种效果称作为徽章效果，使用“badge”样式来实现。

对应的文件版本：

  ☑   LESS版本:源文件badges.less

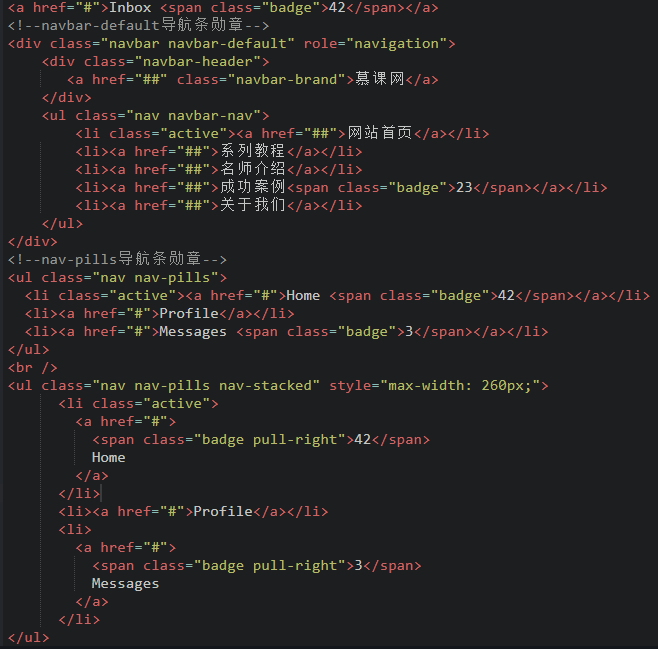
  ☑   Sass版本：源文件\_badges.scss

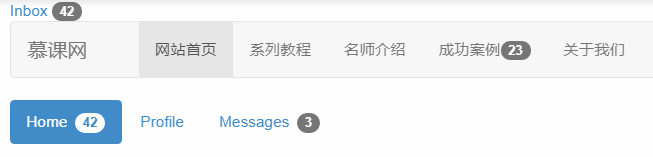
  ☑   编译后版本：bootstrap.css文件第4328行～第4366行

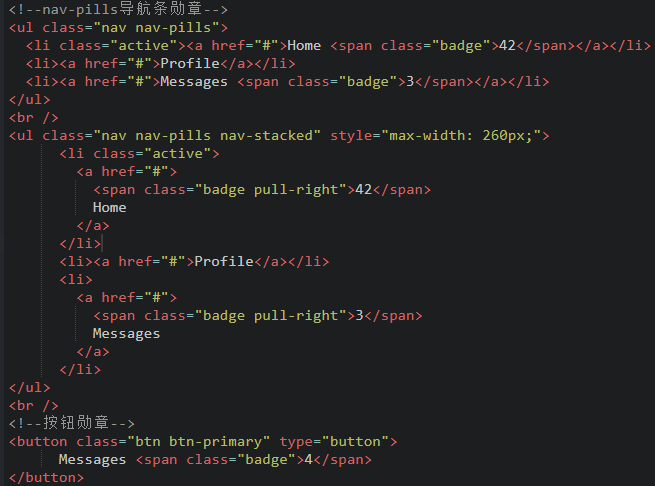
**使用方法：**

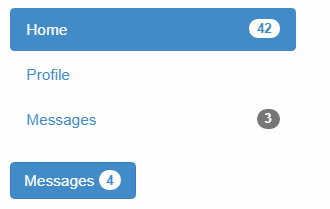
使用方法，其实也没什么太多可说的，你可以像标签一样，使用span标签来制作，然后为他加入badge类：

<a href="#">Inbox <span class="badge">42</span></a>









### 第7章 其它内置组件

在网页制作中还有许多有用的小组件，如列表框、缩略图等，这一章就将给大家尾尾道来。

## 7-1缩略图（一）

**缩略图**在网站中最常用的地方就是产品列表页面，一行显示几张图片，有的在图片底下（左侧或右侧）带有标题、描述等信息。Bootstrap框架将这一部独立成一个模块组件。并通过“**thumbnail**”样式配合**bootstrap的网格系统**来实现。可以将产品列表页变得更好看。

源码文件：

  ☑  LESS版本：对应文件thumbnails.less

  ☑  Sass版本：对应文件\_thumbnails.scss

  ☑ 编译后版本：bootstrap.css文件第4402行～第4426行

**使用方法：**

通过“**thumbnail**”样式配合**bootstrap的网格系统**来实现。

前面也说过了，缩略图的实现是配合网格系统一起使用，假设我们一个产品列表，如下图所示：

[](http://img.mukewang.com/5418e97a00014d6806620159.jpg)

先来看结构：

<div class="container">

<div class="row">

<div class="**col-xs-6 col-md-3**">

<a href="#" class="**thumbnail**">

<img src="http://img.mukewang.com/5434eba100014fe906000338.png" style="height: 180px; width: 100%; display: block;" alt="">

</a>

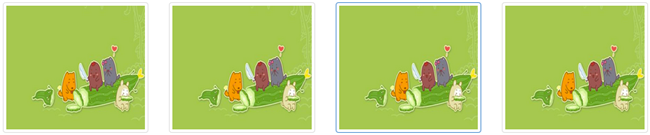
</div>

…

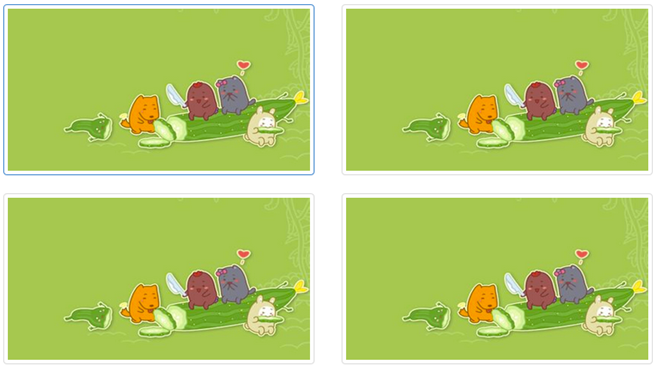
</div>

</div>

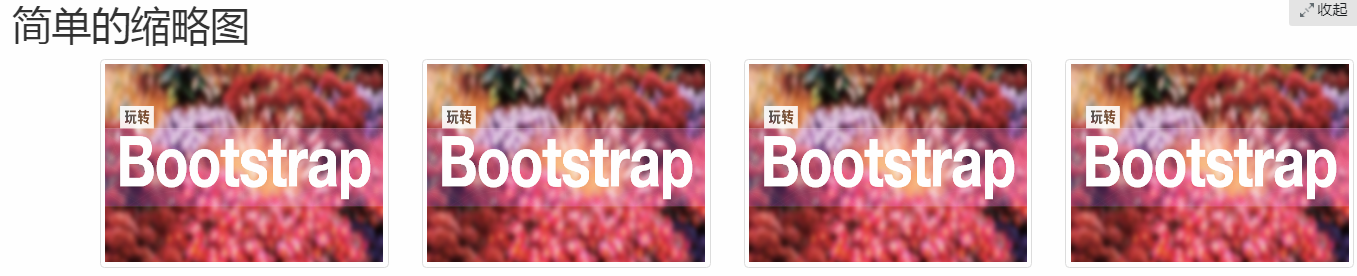
上面的结构表示的是在宽屏幕（可视区域大于768px）的时候，一行显示四个缩略图(单击全屏查看效果)：

[](http://img.mukewang.com/5418ea8a00016d7c06500135.jpg)

在窄屏（可视区域小于768px）的时候，一行只显示两个缩略图：

[](http://img.mukewang.com/5418eac00001bf4a06550366.jpg)







## 缩略图（二）

上一小节，展示的仅只有缩略图的一种使用方式，除了这种方式之外，还可以让缩略图配合标题、描述内容，按钮等：

[](http://img.mukewang.com/5418f3a20001103f06620230.jpg)

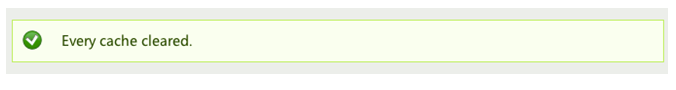
在仅有缩略图的基础上，添加了一个div名为“caption“的容器，在这个容器中放置其他内容，比如说标题，文本描述，按钮等：





## 7-3警示框

在网站中，网页总是需要和用户一起做沟通与交流。特别是当用户操作上下文为用户提供一些有效的警示框，比如说告诉用户操作成功、操作错误、提示或者警告等。如下图所示：

[](http://img.mukewang.com/5418f5120001a99e06760090.jpg)

在Bootstrap框架有一个独立的组件，实现上述的效果，这个组件被称为警示框。

**源码版本：**

☑ LESS版本：对应的源码文件alerts.less

☑ Sass版本：对应的源码文件\_alerts.scss

☑ 编译后的版本：bootstrap.css文件第4427行～第4499行

下面几个小节会对这几种警示框详细讲解，大家不要心急噢！

## 7-4警示框--默认警示框

Bootstrap框架通过“alert“样式来实现警示框效果。在默认情况之下，提供了四种不同的警示框效果：

**1、成功警示框**：告诉用用户操作成功，在“alert”样式基础上追加“alert-success”样式，具体呈现的是背景、边框和文本都是绿色；

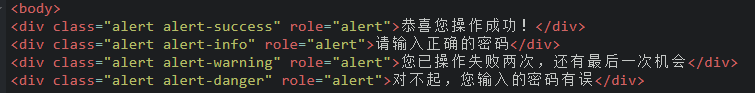
**2、信息警示框**：给用户提供提示信息，在“alert”样式基础上追加“alert-info”样式，具体呈现的是背景、边框和文本都是浅蓝色；

**3、警告警示框**：提示用户小心操作（提供警告信息），在“alert”样式基础上追加“alert-warning”样式，具体呈现的是背景、边框、文本都是浅黄色；

**4、错误警示框**：提示用户操作错误，在“alert”样式基础上追加“alert-danger”样式，具体呈现的是背景、边框和文本都是浅红色。

**使用方法：**

具体使用的时候，可以在类名为“alert”的div容器里放置提示信息。实现不同类型警示框，只需要在“alert”基础上追加对应的类名



## 7-5警示框--可关闭的警示框

大家在平时浏览网页的时候，会发现一些警示框带有关闭按钮，用户一点击关闭按钮就能自动关闭显示的警示框（也就是让警示框隐藏不显示）。在Bootstrap框架中的警示框也具有这样的功能。

**使用方法：**

只需要在默认的警示框里面添加一个关闭按钮。然后进行三个步骤：

  1、需要在基本警示框“alert”的基础上添加“**alert-dismissable**”样式。

  2、在button标签中加入**class="close"**类，实现警示框关闭按钮的样式。

  3、要确保关闭按钮元素上设置了自定义属性：“**data-dismiss="alert"**”（因为可关闭警示框需要借助于Javascript来检测该属性，从而控制警示框的关闭）。

具体使用如下：

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

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

恭喜您操作成功！

</div>

运行效果如下：

[](http://img.mukewang.com/5418f90a0001127e06660238.jpg)



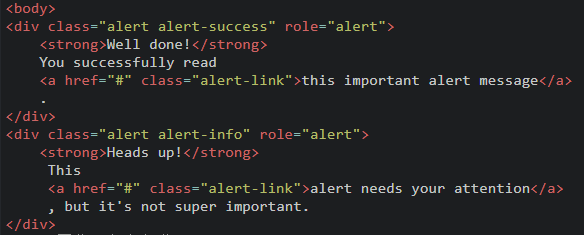
其中“alert”样式的源码主要是设置了警示框的背景色、边框、圆角和文字颜色。另外对其内部几个元素h4、p、ul和“.alert-link”做了样式上的特殊处理，不同类型的警示框，主要是通过“alert-success”、“alert-info”、“alert-warning”和“alert-danger”样式来实现

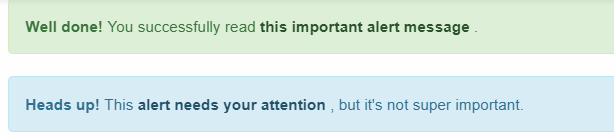
## 7-6警示框--警示框的链接

有时候你可能想在警示框中加入链接地址，用来告诉用户跳到某一个地方或新的页面。而这个时候你又想让用户能明显的看出来这是链接地址。在Bootstrap框架中对警示框里的链接样式做了一个高亮显示处理。为不同类型的警示框内的链接进行了加粗处理，并且颜色相应加深。

**实现方法：**

Bootstrap框架是通过给警示框加的链接添加一个名为“alert-link”的类名，通过“alert-link”样式给链接提供高亮显示。





## 7-7进度条

在网页中，进度条的效果并不少见，比如一个评分系统，比如加载状态等。就如下图所示的一个评分系统，他就是一个简单的进度条效果：

[](http://img.mukewang.com/5418fe4e000106af05640152.jpg)

进度条和其他独立组件一样，开发者可以根据自己的需要，选择对应的版本：

  ☑ LESS版本：源码文件progress-bars.less

  ☑ Sass版本：源码文件\_progress-bars.scss

  ☑ 编译后版本：bootstrap.css文件第4500行～第4575行

而且Bootstrap框架为大家提供多种样式风格的进度条，供大家使用

## 7-8进度条--基本样式

Bootstrap框架中对于进度条提供了一个基本样式，一个100%宽度的背景色，然后个高亮的色表示完成进度。其实制作这样的进度条非常容易，一般是使用两个容器，外容器具有一定的宽度，并且设置一个背景颜色，他的子元素设置一个宽度，比如完成度是30%（也就是父容器的宽度比例值），同时给其设置一个高亮的背景色。

**使用方法：**

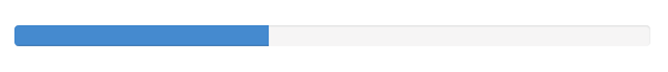
Bootstrap框架中也是按这样的方式实现的，他提供了两个容器，外容器使用“progress”样式，子容器使用“progress-bar”样式。其中progress用来设置进度条的容器样式，而progress-bar用于限制进度条的进度。使用方法非常的简单：

<div class="**progress**">

<div class="**progress-bar**" style="width:40%"></div>

</div>

运行效果如下：

[](http://img.mukewang.com/5418ff230001f9e106680072.jpg)

**结构优化：**

虽然这样实现了基本进度条效果，但对于残障人员浏览网页有点困难，所以我们可以将结构做得更好些（语义化更友好些）：

<div class="progress">

<div class="progress-bar" style="width:40%;" role="progressbar" aria-valuenow="40" aria-valuemin="0" aria-valuemax="100">

<span class="sr-only">40% Complete</span>

</div>

</div>

1、role属性作用：告诉搜索引擎这个div的作用是进度条。

2、aria-valuenow="40"属性作用：当前进度条的进度为40%。

3、aria-valuemin="0"属性作用：进度条的最小值为0%。

4、aria-valuemax="100"属性作用：进度条的最大值为100%。

## 7-9进度条--彩色进度条

Bootstrap框架中的进度条和警告信息框一样，为了能给用户一个更好的体验，也根据不同的状态配置了不同的进度条颜色。在此称为彩色进度条，其主要包括以下四种：

  ☑ **progress-bar-info**：表示信息进度条，进度条颜色为蓝色

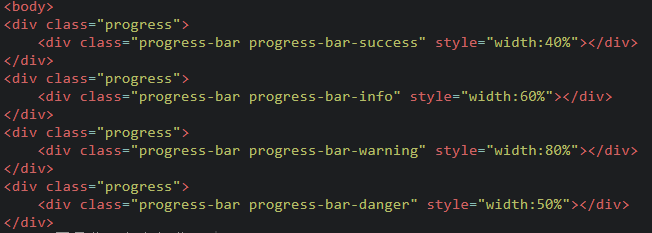
  ☑ **progress-bar-success**：表示成功进度条，进度条颜色为绿色

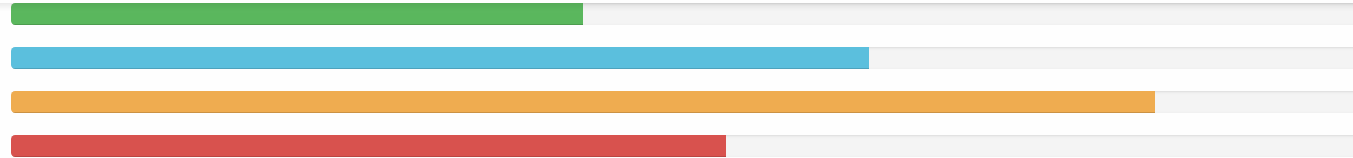
  ☑ **progress-bar-warning**：表示警告进度条，进度条颜色为黄色

  ☑ **progress-bar-danger**：表示错误进度条，进度条颜色为红色

**使用方法：**

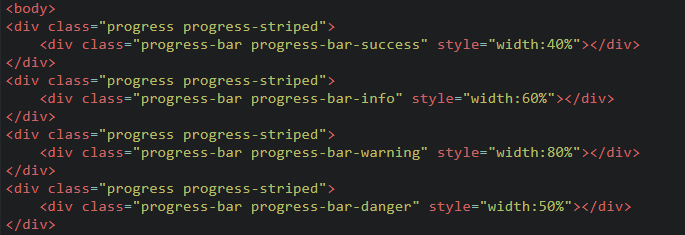
具体使用就非常简单了，只需要在基础的进度上增加对应的类名。

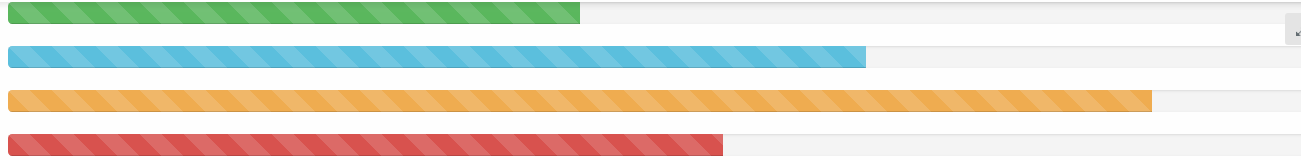




## 7-10进度条--条纹进度条

在Bootstrap框架中除了提供了彩色进度条之外，还提供了一种条纹进度条，这种条纹进度条采用CSS3的线性渐变来实现，并未借助任何图片。使用Bootstrap框架中的条纹进度条只需要在进度条的容器“progress”基础上增加类名“progress-striped”，当然，如果你要让你的进度条条纹像彩色进度一样，具有彩色效果，你只需要在进度条上增加相应的颜色类名，如前面的彩色进度条所讲。





## 7-11进度条--动态条纹进度条

**使用方法：**

在进度条“progress progress-striped”两个类的基础上再加入“**active**”类名。如下代码：

<div class="progress progress-striped **active**">

<div class="progress-bar progress-bar-success" style="width:40%"></div>

</div>

**特别注意**：要让条纹进度条动起来，就需要让“progress-striped”和“active”同时运用，不然条纹进度条是不具备动效效果

## 7-12进度条--层叠进度条

Bootstrap框架除了提供上述几种进度条之外，还提供了一种层叠进度条，层叠进度条，可以将不同状态的进度条放置在一起，按水平方式排列。具体使用如下：

<div class="progress">

<div class="progress-bar progress-bar-success" style="width:20%"></div>

<div class="progress-bar progress-bar-info" style="width:10%"></div>

<div class="progress-bar progress-bar-warning" style="width:30%"></div>

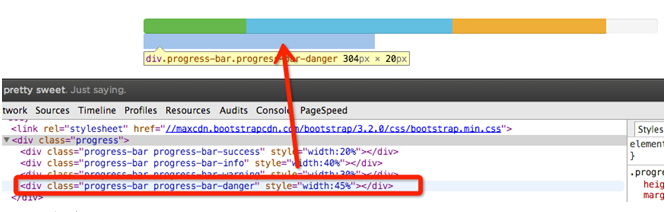
<div class="progress-bar progress-bar-danger" style="width:15%"></div>

</div>

运行效果如下：

[](http://img.mukewang.com/5419242a0001b6cc06660074.jpg)

或许你会感到疑问，没有为层叠进度条添加额外的样式代码，怎么就正常了呢？可以回过头来看基本进度条那部分，不难发现，在“progress-bar”上有一个左浮动的样式。也就是这个样式，在不增加任何样式代码就能实现上例的层叠效果。当然有一点需要注意，层叠进度条宽度之和不能大于100%，大于100%就会造成下面的不良效果（见右侧代码编辑器）：

[](http://img.mukewang.com/541924530001ed6e06650212.jpg)

除了层叠彩色进度条之外，还可以层叠条纹进度条，或者说条纹进度条和彩色进度条混合层叠，仅需要在“progress”容器中添加对应的进度条，同样要注意，层叠的进度条之和不能大于100%

## 7-13进度条--带Label的进度条

上面介绍的各种进度条，都仅仅是通过颜色进度向用户传递进度值。但实际中，有很多时候是需要在进度条中直接用相关的数值向用户传递完成的进度值，在Bootstrap就为大家考虑了这种使用场景。

**实现方法：**

只需要在进度条中添加你需要的值，如：

<div class="progress">

<div class="progress-bar progress-bar-success" role="progressbar" aria-valuenow="20" aria-valuemin="0" aria-valuemax="100" style="width:20%">**20%**</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541928420001baea06610148.jpg)

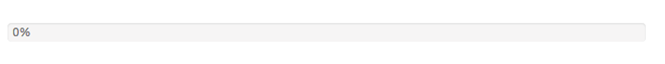
还有一种特殊情形，当进度条处于开始位置，也就是进度条的值为0%时，内容是否会撑开一定的宽度，让进度条具有颜色呢？如果是，这不是我们需要的效果，如果不是，又是怎么实现的呢？我们先来看一个这样的示例：

<div class="progress">

<div class="progress-bar" role="progressbar" aria-valuenow="0" aria-valuemin="0" aria-valuemax="100">0%</div>

</div>

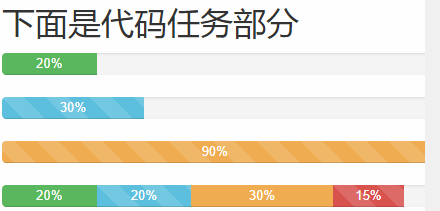
运行效果如下：

[](http://img.mukewang.com/541928ab0001b41506650064.jpg)



综合实例：





## 7-14媒体对象

在Web页面或者说移动页面制作中，常常看到这样的效果，左边居左（或居右），内容居右（或居左）排列，如下图所示：

[](http://img.mukewang.com/54192a4d00014b6a06590154.jpg)

我们常常把这样的效果称为媒体对象。可以说他是一种抽像的样式，可以用来构建不同类型的组件。这些组件都具有开篇所说的样式风格。那么在Bootstrap框架中特意将些部分提取出来做来一个组件介绍。其对应的版本文件：

  ☑**LESS版本**：对应的源文件是media.less

  ☑**Sass版本：**对应的源文件是\_media.scss

  ☑**编译后版本：**对应bootstrap.css文件第4792行～第4819行

## 7-15媒体对象--默认媒体对象

媒体对象一般是成组出现，而一组媒体对象常常包括以下几个部分：

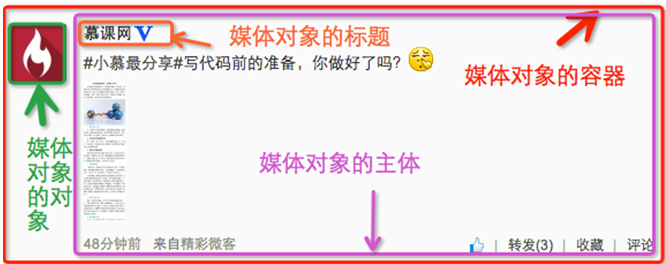
  ☑   媒体对像的容器：常使用“media”类名表示，用来容纳媒体对象的所有内容

  ☑  媒体对像的对象：常使用“media-object”表示，就是媒体对象中的对象，常常是图片

  ☑  媒体对象的主体：常使用“media-body”表示，就是媒体对像中的主体内容，可以是任何元素，常常是图片侧边内容

  ☑  媒体对象的标题：常使用“media-heading”表示，就是用来描述对象的一个标题，此部分可选

如下图所示：

[](http://img.mukewang.com/54192bd200016f6306660264.jpg)

除了上面四个部分之外，在Bootstrap框架中还常常使用“pull-left”或者“pull-right”来控制媒体对象中的对象浮动方式。

在具体使用中如下所示：

<div class="**media**">

<a class="**pull-left**" href="#">

<img class="**media-object**" src="http://img.mukewang.com/52e1d29d000161fe06000338-300-170.jpg" alt="...">

</a>

<div class="**media-body**">

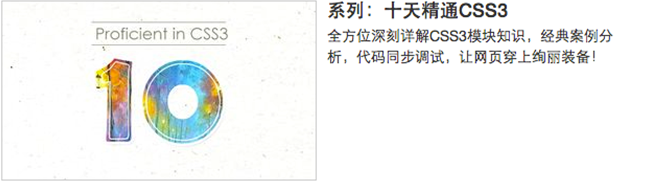
<h4 class="**media-heading**">系列：十天精通CSS3</h4>

<div>全方位深刻详解CSS3模块知识，经典案例分析，代码同步调试，让网页穿上绚丽装备！</div>

</div>

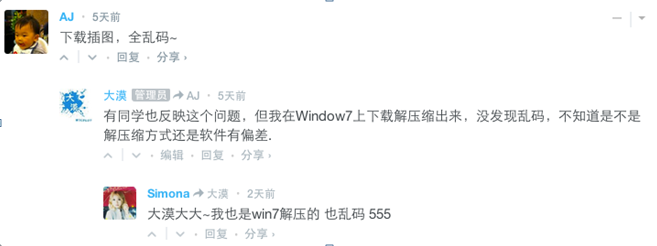
</div>

运行效果如下：

[](http://img.mukewang.com/54192c430001e72b06540183.jpg)

## 7-16媒体对象--媒体对象的嵌套

在评论系统中，常常能看到下图的效果：

[](http://img.mukewang.com/54192d740001e45706570246.jpg)

从外往里看，这里有三个媒体对象，只不过是一个嵌套在另一个的里面。那么在Bootstrap框架中的媒体对象也具备这样的功能，只需要将另一个媒体对象结构放置在媒体对象的主体内“media-body”





## 7-17媒体对象--媒体对象列表

媒体对象的嵌套仅是媒体对象中一个简单应用效果之一，在很多时候，我们还会碰到一个列表，每个列表项都和媒体对象长得差不多，同样用评论系统来说事：

[](http://img.mukewang.com/541930820001e33e06580376.jpg)

**使用方法：**

针对上图的媒体对象列表效果，Bootstrap框架提供了一个列表展示的效果，在写结构的时候可以使用ul，并且在ul上添加类名“media-list”，而在li上使用“media”，示例代码如下：

<ul class="**media-list**">

<li class="**media**">

<a class="pull-left" href="#">

<img class="media-object" src=" " alt="...">

</a>

<div class="media-body">

<h4 class="media-heading">Media Header</h4>

<div>…</div>

</div>

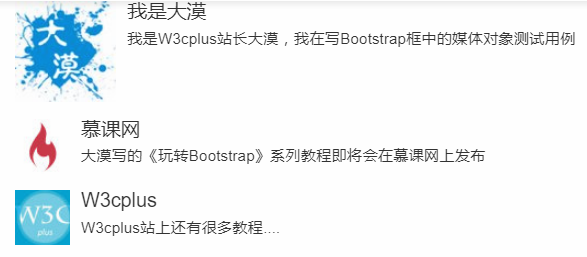
</li>

<li class="**media**">…</li>

<li class="**media**">…</li>

</ul>





## 7-18列表组

列表组是Bootstrap框架新增的一个组件，可以用来制作列表清单、垂直导航等效果，也可以配合其他的组件制作出更漂亮的组件。由于其在Bootstrap是一个独立的组件，所以也对应有自己独立源码：

  ☑**LESS版本**：对应的源码文件 list-group.less

  ☑   **Sass版本**：对应的源码文件是 \_list-group.scss

  ☑   **编译出的Bootstrap版本：**对应的源码bootstrap.css文件第4820行～第4994行

## 7-19列表组--基础列表组

基础列表组，看上去就是去掉了列表符号的列表项，并且配上一些特定的样式。在Bootstrap框架中的基础列表组主要包括两个部分：

  ☑  **list-group**：列表组容器，常用的是ul元素，当然也可以是ol或者div元素

  ☑  **list-group-item**：列表项，常用的是li元素，当然也可以是div元素

来看一个简单的示例：

<ul class="**list-group**">

<li class="**list-group-item**">揭开CSS3的面纱</li>

<li class="**list-group-item**">CSS3选择器</li>

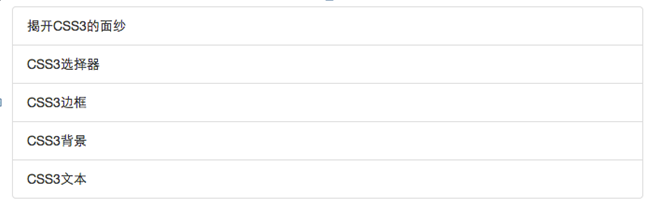
<li class="**list-group-item**">CSS3边框</li>

<li class="**list-group-item**">CSS3背景</li>

<li class="**list-group-item**">CSS3文本</li>

</ul>

运行效果如下：

[](http://img.mukewang.com/541938050001335e06550202.jpg)

## 7-20列表组--带徽章的列表组

带徽章的列表组其实就是将Bootstrap框架中的徽章组件和基础列表组结合在一起的一个效果。具体做法很简单，只需要在“list-group-item”中添加徽章组件“badge”：

<ul class="list-group">

<li class="list-group-item">

**<span class="badge">13</span>**揭开CSS3的面

</li>

<li class="list-group-item">

**<span class="badge">456</span>**CSS3选择器

</li>

<li class="list-group-item">

**<span class="badge">892</span>**CSS3边框

</li>

<li class="list-group-item">

**<span class="badge">90</span>**CSS3背景

</li>

<li class="list-group-item">

**<span class="badge">1290</span>**CSS3文本

</li>

</ul>

运行效果如下：

[](http://img.mukewang.com/541939cb000111e006550205.jpg)

## 7-21列表组--带链接的列表组

带链接的列表组，其实就是每个列表项都具有链接效果。大家可能最初想到的就是在基础列表组的基础上，给列表项的文本添加链接：

<ul class="list-group">

<li class="list-group-item">

<a href="##">揭开CSS3的面</a>

</li>

<li class="list-group-item">

<a href="##">CSS3选择器</a>

</li>

...

</ul>

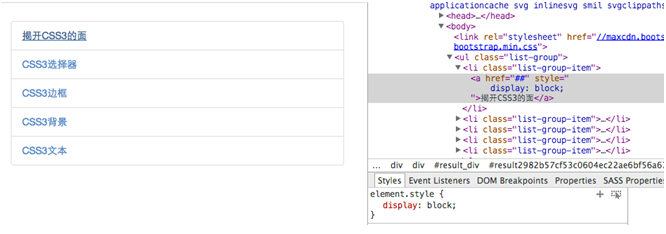
运行效果如下：

[](http://img.mukewang.com/54193e2b0001e6e606470202.jpg)

这样做有一个不足之处，就是链接的点击区域只在文本上有效：

[](http://img.mukewang.com/54193e7300017f9406590195.jpg)

但很多时候，都希望在列表项的任何区域都具备可点击。这个时候就需要在链接标签上增加额外的样式：“display:block”；

[](http://img.mukewang.com/54193eba00018ae306660234.jpg)

虽然这样能解决问题，达到需求。但在Bootstrap框架中，还是采用了另一种实现方式。就是将ul.list-group使用div.list-group来替换，而li.list-group-item直接用a.list-group-item来替换。这样就可以达到需要的效果：

<**div** class="list-group">

<**a** href="##" class="list-group-item">图解CSS3<**/a**>

<**a** href="##" class="list-group-item"><span class="badge">220</span>Sass教程<**/a**>

<**a** href="##" class="list-group-item">玩转Bootstrap<**/a**>

<**/div**>

运行效果如下：

[](http://img.mukewang.com/54193f4d0001e76b06570137.jpg)

## 7-22列表组--自定义列表组

Bootstrap框加在链接列表组的基础上新增了两个样式：

  ☑  list-group-item-heading：用来定义列表项头部样式

  ☑  list-group-item-text：用来定义列表项主要内容

这两个样式最大的作用就是用来帮助开发者可以自定义列表项里的内容，如下面的示例：

<div class="list-group">

<a href="##" class="list-group-item">

<h4 class="**list-group-item-heading**">图解CSS3</h4>

<p class="**list-group-item-text**">...</p>

</a>

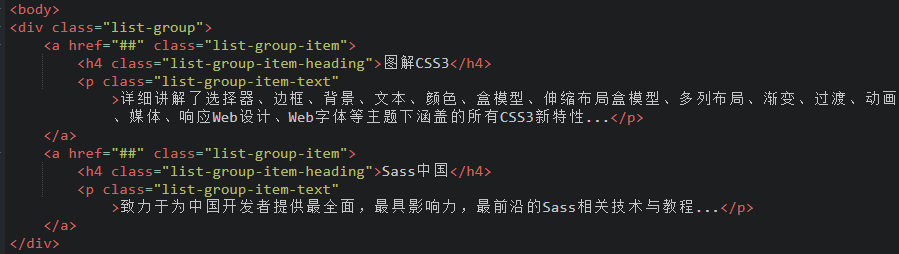
<a href="##" class="list-group-item">

<h4 class="**list-group-item-heading**">Sass中国</h4>

<p class="**list-group-item-text**">...</p>

</a>

</div>





## 7-23列表组--列表项的状态设置

Bootstrap框架也给组合列表项提供了状态效果，特别是链接列表组。比如常见状态和禁用状态等。实现方法和前面介绍的组件类似，在列表组中只需要在对应的列表项中添加类名：

  ☑  **active**：表示当前状态

  ☑  **disabled**：表示禁用状态

来看个示例：

<div class="list-group">

<a href="##" class="list-group-item **active**"><span class="badge">5902</span>图解CSS3</a>

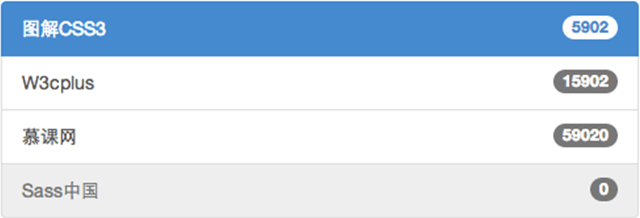
<a href="##" class="list-group-item"><span class="badge">15902</span>W3cplus</a>

<a href="##" class="list-group-item"><span class="badge">59020</span>慕课网</a>

<a href="##" class="list-group-item **disabled**"><span class="badge">0</span>Sass中国</a>

</div>

运行效果如下：

[](http://img.mukewang.com/5419437d0001b70006400218.jpg)

## 7-24列表组--多彩列表组

列表组组件和警告组件一样，Bootstrap为不同的状态提供了不同的背景颜色和文本色，可以使用这几个类名定义不同背景色的列表项。

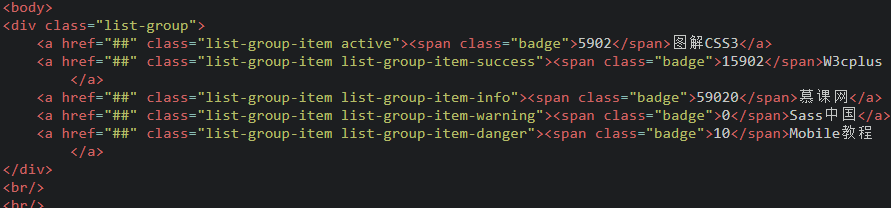
  ☑  list-group-item-success：成功，背景色绿色

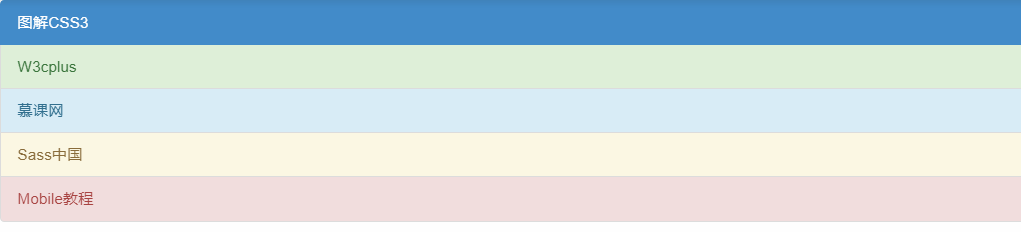
  ☑  list-group-item-info：信息，背景色蓝色

  ☑  list-group-item-warning：警告，背景色为黄色

  ☑  list-group-item-danger：错误，背景色为红色

如果你想给列表项添加什么背景色，只需要在“list-group-item”基础上增加对应的类名





## 7-25面板

面板（Panels）是Bootstrap框架新增的一个组件，其主要作用就是用来处理一些其他组件无法完成的功能。同样在不同的版本中具有不同的源码：

  ☑  Less版本：对应的源码文件是 panels.less

  ☑  Sass版本：对应的源码文件是 \_panels.scss

  ☑  编译后的Bootstrap：对应bootstrap.css文件第4995行～第5302行

## 7-26面板--基础面板

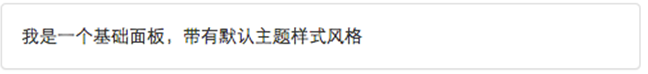
基础面板非常简单，就是一个div容器运用了“panel”样式，产生一个具有边框的文本显示块。由于“panel”不控制主题颜色，所以在“panel”的基础上增加一个控制颜色的主题“panel-default”，另外在里面添加了一个“div.panel-body”来放置面板主体内容：

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

<div class="**panel-body**">我是一个基础面板，带有默认主题样式风格</div>

</div>

运行效果如下：

[](http://img.mukewang.com/54194ead00016f8806480074.jpg)

**原理分析：**

“panel“主要对边框，间距和圆角做了一定的设置

## 7-27面板--带有头和尾的面板

基础面板看上去太简单了，Bootstrap为了丰富面板的功能，特意为面板增加“面板头部”和“页面尾部”的效果：

   ☑  panel-heading：用来设置**面板头部**样式

   ☑ panel-footer：用来设置**面板尾部**样式

<div class="panel panel-default">

<div class="panel-heading">图解CSS3</div>

<div class="panel-body">…</div>

<div class="panel-footer">作者：大漠</div>

</div>

运行效果如下：

[](http://img.mukewang.com/5419500b00017cc706440219.jpg)

**原理分析：**

panel-heading和panel-footer也仅仅间距和圆角等样式进行了设置

## 7-28面板--彩色面板

在基础面板一节中了解到，panel样式并没有对主题进行样式设置，而主题样式是通过panel-default来设置。在Bootstrap框架中面板组件除了默认的主题样式之外，还包括以下几种主题样式，构成了一个彩色面板：

  ☑  panel-primary：重点蓝

  ☑  panel-success：成功绿

  ☑ panel-info:信息蓝

  ☑ panel-warning：警告黄

  ☑ panel-danger：危险红

使用方法就很简单了，只需要在panel的类名基础上增加自己需要的类名：

<div class="panel panel-default">

<div class="panel-heading">图解CSS3</div>

<div class="panel-body">…</div>

<div class="panel-footer">作者：大漠</div>

</div>

<div class="panel panel-primary">…</div>

<div class="panel panel-success">…</div>

<div class="panel panel-info">…</div>

<div class="panel panel-warning">…</div>

<div class="panel panel-danger">…</div>

运行效果如下：



从效果中不难发现，这几个样式只是改变了面板的背景色、文本和边框颜色

## 7-29面板--面板中嵌套表格

一般情况下可以把面板理解为一个区域，在使用面板的时候，都会在panel-body放置需要的内容，可能是图片、表格或者列表等。来看看面板中嵌套表格和列表组的一个效果。首先来看嵌套表格的效果：

<div class="panel panel-default">

<div class="panel-heading">图解CSS3</div>

<div class="panel-body">

<p>详细讲解了选择器、边框、背景、文本、颜色、盒模型、伸缩布局盒模型、多列布局、渐变、过渡、动画、媒体、响应Web设计、Web字体等主题下涵盖的所有CSS3新特性

</p>

<table class="table table-bordered">

<thead>

<tr>

<th>＃</th>

<th>我的书</th>

<th>发布时间</th>

</tr>

</thead>

<tbody>

<tr>

<td>1</td>

<td>《图解CSS3》</td>

<td>2014-07-10</td>

</tr>

</tbody>

</table>

</div>

<div class="panel-footer">作者：大漠</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541954b90001781f06520370.jpg)

**优化代码：**

在实际应用运中，你或许希望表格和面板边缘不需要有任何的间距。但由于panel-body设置了一个padding：15px的值，为了实现这样的效果。我们在实际使用的时候需要把table提取到panel-body外面：

<div class="panel panel-default">

<div class="panel-heading">图解CSS3</div>

<div class="panel-body">…</div>

<table class="table table-bordered">…</table>

<div class="panel-footer">作者：大漠</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541955440001d27206510331.jpg)

这样的效果是不是完美多了。大家可能会问，前面介绍表格的时候table-bordered明明有边框，按理说这里应该会出现边框重叠效果才对，怎么没有呢？其实原本是有边框重叠的，只不过在面板中对表格又做了一次优化

## 面板--面板中嵌套列表组

在上一节，我们介绍了如何在面板中放置表格，现在我们来学习如何在面板中放置列表组，我们简单的来看一个示例：

<div class="panel panel-default">

<div class="panel-heading">图解CSS3</div>

<div class="panel-body">

<p>详细讲解了选择器、边框、背景、文本、颜色、盒模型、伸缩布局盒模型、多列布局、渐变、过渡、动画、媒体、响应Web设计、Web字体等主题下涵盖的所有CSS3新特性

</p>

<ul class="list-group">

<li class="list-group-item">我是列表项</li>

<li class="list-group-item">我是列表项</li>

<li class="list-group-item">我是列表项</li>

</ul>

</div>

<div class="panel-footer">作者：大漠</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541957920001280d06600435.jpg)

**优化代码：**

和嵌套表格一样，如果你觉得这样有间距不好看，你完全可以把列表组提取出来：

<div class="panel panel-default">

<div class="panel-heading">图解CSS3</div>

<div class="panel-body">…</div>

<ul class="list-group">

<li class="list-group-item">我是列表项</li>

<li class="list-group-item">我是列表项</li>

<li class="list-group-item">我是列表项</li>

</ul>

<div class="panel-footer">作者：大漠</div>

</div>

运行效果如下：

[](http://img.mukewang.com/541957ed000127f106440360.jpg)

同样的道理，Bootstrap将嵌套在面板中的列表组做了一定的样式优化。

### 第8章 Bootstrap支持的JavaScript插件

本章节讲Bootstrap支持的JavaScript插件,包括选顶卡、弹出框、提示框。

## 导入JavaScript插件

Bootstrap除了包含丰富的Web组件之外，如前面介绍的下拉菜单、按钮组、导航、分页等。他还包括一些JavaScript的插件。

Bootstrap的JavaScript插件可以单独导入到页面中，也可以一次性导入到页面中。因为在Bootstrap中的JavaScript插件都是依赖于jQuery库，所以不论是单独导入还一次性导入之前必须先导入jQuery库。

**一次性导入：**

Bootstrap提供了一个单一的文件，这个文件包含了Bootstrap的所有JavaScript插件，即bootstrap.js（压缩版本：bootstrap.min.js）。

具体使用如下（或见右侧代码编辑器28-29行）：

<!—导入jQuery版本库，因为Bootstrap的JavaScript插件依赖于jQuery -->

<script src="http://libs.baidu.com/jquery/1.9.0/jquery.js"></script>

<!—- 一次性导入所有Bootstrap的JavaScript插件（压缩版本） -->

<script src="js/bootstrap.min.js"></script>

特别声明：jQuery版本库也可以加载你本地的jQuery版本。

**单独导入：**

为方便单独导入特效文件，Bootstrap V3.2中提供了12种JavaScript插件，他们分别是：

**☑** **动画过渡（Transitions）:**对应的插件文件“transition.js”

**☑** **模态弹窗（Modal）:**对应的插件文件“modal.js”

**☑** **下拉菜单（Dropdown）：**对应的插件文件“dropdown.js”

**☑** **滚动侦测（Scrollspy）：**对应的插件文件“scrollspy.js”

**☑** **选项卡（Tab）：**对应的插件文件“tab.js”

**☑** **提示框（Tooltips）：**对应的插件文件“tooltop.js”

**☑** **弹出框（Popover）：**对应的插件文件“popover.js”

**☑** **警告框（Alert）：**对应的插件文件“alert.js”

**☑** **按钮（Buttons）：**对应的插件文件“button.js”

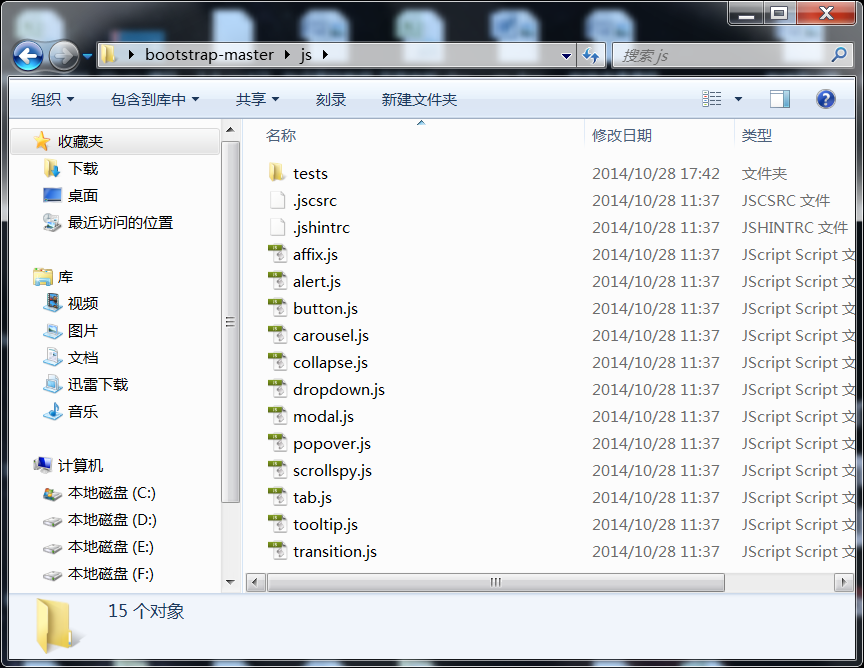
**☑** **折叠/手风琴（Collapse）：**对应的插件文件“collapse.js”

**☑** **图片轮播Carousel：**对应的插件文件“carousel.js”

**☑** **自动定位浮标Affix：**对应的插件文件“affix.js”

上述单独插件的下载可到github去下载（<https://github.com/twbs/bootstrap>）。

下载后可查看js文件夹，如下图：

[](http://img.mukewang.com/544f639a0001116608640668.jpg)

接下来依次向大家介绍这些插件如何使用，至于插件源码分析，在本系列不做过多的阐述，具体源码可以阅读各插件的代码。

**注意：**在后面的例子中我们为了方便都采用**一**次性导入的方法，即引入“bootstrap.min.js”文件（小伙伴们可以在自己的项目中跟据需要选择单独导入还是一次性导入）。

## 8-2动画过渡（Transitions）

这一小节我们先来讲“动画过渡（Transitions）”这个插件的使用，源文件：transition.js

Bootstrap框架默认给各个组件提供了基本动画的过渡效果，如果要使用，有两种方法：

  ☑ 调用统一编译的bootstrap.js；

  ☑ 调用单一的过渡动画的JavaScript插件文件transition.js（右侧第29行引入Bootstrap上对外公布的transition.js的地址）。

transition.js文件为Bootstrap具有过渡动画效果的组件提供了动画过渡效果。不过需要注意的是，这些过渡动画都是采用CSS3来实现的，所以IE6-8浏览器是不具备这些过渡动画效果。

默认情况之下，Bootstrap框架中以下组件使用了过渡动画效果：

  ☑ 模态弹出窗（Modal）的滑动和渐变效果；

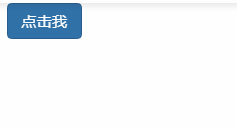
  ☑ 选项卡（Tab）的渐变效果；

  ☑ 警告框（Alert）的渐变效果；

  ☑ 图片轮播（Carousel）的滑动效果。

举了一个“模态弹出窗（Modal）的滑动和渐变效果”源代码例子。



## 8-3模态弹出框（Modals）

这一小节我们先来讲解一个“模态弹出框”，插件的源文件：modal.js。

右侧代码编辑器（30行）就是单独引入 bootstrap 中发布出的“modal.js”文件。

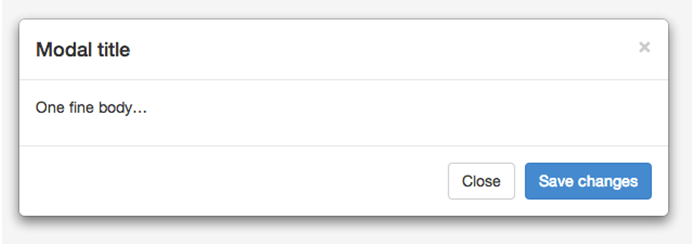
样式代码：

  ☑ LESS版本：modals.less

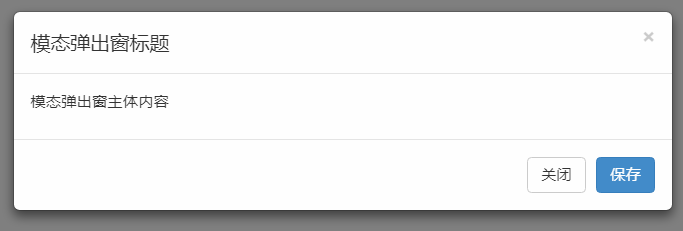
  ☑ Sass版本：\_modals.scss

  ☑ 编译后的Bootstrap：对应bootstrap.css文件第5375行～第5496行

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。一般用于弹出提示信息，确认信息，表单等内容，如下图所示：

[](http://img.mukewang.com/541a3ecc00016cc606940246.jpg)



## 8-4模态弹出框--结构分析

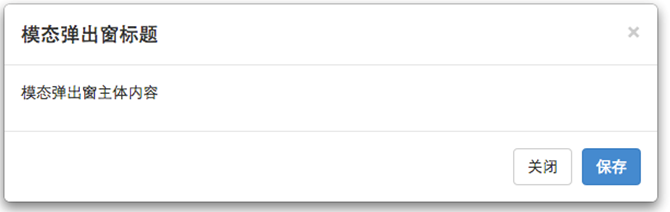
Bootstrap框架中的模态弹出框，分别运用了“modal”、“modal-dialog”和“modal-content”样式，而弹出窗真正的内容都放置在“modal-content”中，其主要又包括三个部分：

**☑ 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

如下图所示：

[](http://img.mukewang.com/541a3f4f000177d406700212.jpg)

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

除此之外，modal-content中的modal-header、modal-body和modal-footer三个部分样式设置：

这三个部分主要控制一些间距的样式。而modal-footer都是用来放置按钮，所以底部还对包含的按钮做了一定的样式处理

## 8-5模态弹出框--实现原理解析（一）

**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。

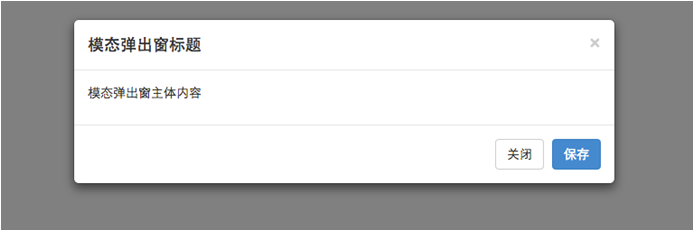
2、单击右侧全屏按钮，在全屏状态下，模态弹出窗宽度是自适应的，而且modal-dialog水平居中。

3、当浏览器视窗大于768px时，模态弹出窗的宽度为600px。

## 8-6模态弹出框--实现原理解析（二）

**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

同样，给其添加了一个过渡动画，从fade到in，把opacity值从0变成了0.5。上图展示的就是in状态下的效果

**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">

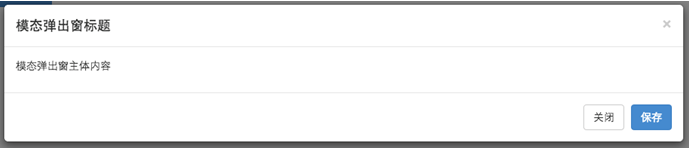
<divclass="modal-dialog **modal-sm**">

<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)

对于这两种尺寸的模态弹出窗，Bootstrap在媒体查询中做过处理

## 8-7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)

## 8-8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>

## 8-9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：

## 8-10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：

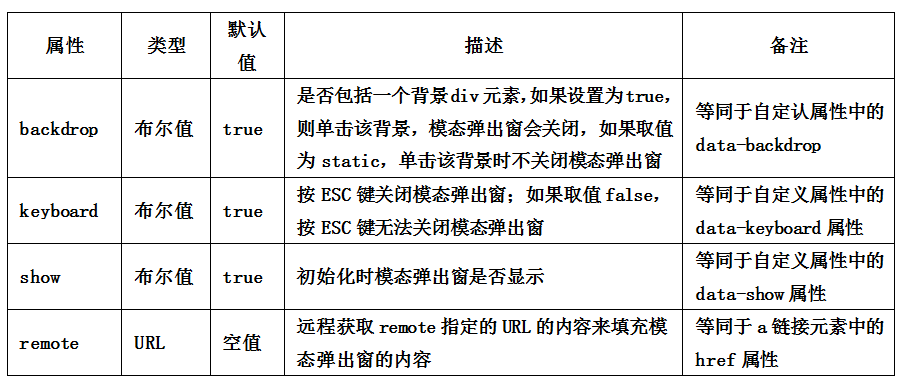


## 8-11模态弹出框--JavaScript触发时的参数设置（一）

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

});

});

});

这一节，我们对JavaScript触发模态弹出窗的属性设置进行了介绍，下一节我们将继续对参数设置和事件设置进行介绍。

## 8-12模态弹出框--JavaScript触发时的参数设置（二）

上一节，我们对JavaScript触发模态弹出窗的属性设置进行了介绍，现在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...

})

## 3-9 标签页制作

