

目 录

简介	9
初学者模板	10
栅格系统	16
机制原理	16
栅格选项	18
自动布局列	18
等宽布局	18
设置一列宽度	19
可变宽度的弹性空间	20
等宽多行	21
响应式的 class 选择器	21
覆盖所有设备	21
水平堆砌	22
混合布局	22
对齐	23
垂直对齐示例	23
水平对齐示例	25
间隙沟槽(gutters)清除	26
列换行	27
换行	27
重排序	28
Class 顺序重定义	28
列偏移	29
Margin 移动布局	30
列嵌套	31
Sass mixins	31
变量	31
Mixins	32
应用案例	32
自定义栅格系统	33
自定义列和间隙	33
自定义栅格	33
初始化与 CSS 重置	35
标题和段落	36
列表	36
pre 预先格式化文本	37
表格	37
Forms 表单	37
其它杂项	38
Address 地址控件	38
Blockquote 引用块效果	38
addr 内联元素	38
概要	39
排版	39
全局设置	39

标题	40
自定义标题备注	41
显式标题	41
Lead 中心内容	41
文本内联元素	41
文本实用程序	42
abbr 缩略语	42
blockquote 来源备注与引用	42
底部备注来源	43
对齐处理	43
列表	43
列表样式初始化	43
分行或单行多列并排	44
dl 表格式水平描述	45
响应式排版	46
代码	46
内联代码	46
代码块	47
Var 变量	47
用户输入（键盘动作提示）	47
示例标注	47
图片	48
响应式图片	48
缩略图处理	48
图像对齐处理	48
Html 5 标准之 Picture 元素	49
表格	49
示例	49
Head 表头处理	51
条纹状表格	53
Bordered table	55
行悬停效果	57
紧缩表格	59
语义状态化	60
Captions 表格辅助标题	62
响应式表格	63
多屏幕断点设定	64
图文框	66
示例	66
文字对齐控制	67
警告提示框(Alert)	67
示例	67
链接颜色	68
额外附加内容	69
关闭警告（小贴士效果）	70
JavaScript 行为	70
触发	70
方法	70
事件	71

徽章(Badge).....	71
示例	71
情景变化	72
椭圆形胶囊标签	72
链接它	73
面包屑导航(Breadcrumb).....	73
无障碍处理	74
按钮(Button).....	74
示例	74
按钮标签	75
轮廓按钮	75
尺寸规格与大小定义	76
启用状态	76
禁用状态	76
按钮插件	77
切换状态	77
复选框和单选框	78
方法	79
按钮组(Btn-group)	80
基本示例	80
按钮工具栏	80
大小规格和尺寸缩放	82
嵌套	82
垂直排列	83
.card 卡片组件（样式）	83
关于	83
示例	83
Card title	83
内容类型	84
主体	84
标题、文字和链接	84
图片	85
列表组	86
混合样式	87
页眉页脚	87
缩放	90
使用栅格系统	90
使用通用全局属性	90
自定义 CSS	91
文本对齐	92
导航	94
图片	95
图片覆盖	95
图像叠加覆盖	96
卡片样式	96
背景和颜色	97
边框	99
Mixins 实用程序	101
卡片排版	101

卡片组	101
Card decks 卡片阵列	104
多列卡片浮动排版	106
轮播效果(Carousel).....	109
工作原理	109
示例	109
经典幻灯片效果	109
带控制器的效果	109
包含姿态指示器	110
包含字幕的轮播	111
用法	112
通过数据属性	112
通过 JavaScript.....	112
选项	112
方法	113
.carousel('dispose')	114
事件	114
折叠面板(Collapse)	115
示例	115
多目标控制	115
手风琴折叠范例	116
无障碍浏览提示(易用性).....	119
用法	119
利用 data 数据属性	120
利用 JavaScript.....	120
选项	120
方法	120
.collapse('dispose')	121
事件	121
下拉菜单 (Dropdowns).....	122
概览	122
无障碍浏览提示(易用性).....	122
示例	122
单一按钮的下拉菜单	122
分裂式按钮下拉菜单	124
尺寸大小定义	124
变形-向上展开	126
Dropright 右指向下拉菜单.....	126
Dropleft 左指向下拉菜单	127
菜单项	128
菜单对齐	129
菜单标题	129
菜单分隔与分割线	129
菜单表单	130
有效菜单项	132
禁用菜单项	132
用例	132
通过事件属性	133
通过 JavaScript.....	133

选项	133
方法	133
事件	133
表单(Forms).....	135
概览	135
表单控件	135
大小规格	137
只读属性	138
只读纯文本	138
复选框与单选框	139
默认堆叠	139
水平排列	140
没有标签	141
布局	141
表单组	141
表单栅格排列	142
垂直排列表单	144
垂直排列表单尺寸规格定义	146
栅格式列尺寸定义	147
内联式表单	149
隐藏的标签替代	151
表单下方帮助提示文本	151
禁用表单	152
验证	153
运行原理	153
自定义样式	153
浏览器默认值	156
服务器端	157
支持元素	159
提示	160
自定义表单	162
复选框和单选框自定义	162
Select menu 下拉选择菜单	165
文件浏览（文件选取）器	166
input 输入框及输入框群组(Input-group)	168
基本示例	168
规格尺寸定义	169
勾选或单选框组合	170
多个输入	170
多类型控件组合	171
按钮组合	171
带下拉列表的按钮组合	172
分裂式按钮与 input 组合	173
自定义表单	174
自定义选择	174
自定义文件输入	175
无障碍浏览提示(易用性)	176
Hero 广告大块屏幕(Jumbotron)	177
列表组(List-group).....	179

基本示例	179
.active 列表(启用)状态指示	179
.disabled 列表(禁用)状态指示	180
链接和按钮	180
Flush 紧致贴齐	182
上下文语境颜色呈现样式	182
引入 badge 徽章	184
自定义内容	185
JavaScript 行为	186
使用数据属性	187
通过 JavaScript	188
fade 淡入淡出效果	188
方法	188
Events 事件	189
.nav 导航/滑动门(nav)	190
基本导航样式	190
可用样式	190
水平对齐	191
垂直排列	191
Tabs 标签	192
胶囊式标签页	193
填充和对齐	193
使用 Flex 弹性布局	195
无障碍易用性处理	195
使用下拉菜单	195
带下拉列表的标签页	195
带下拉列表的胶囊式标签页	196
JavaScript 行为/滑动门	197
使用数据属性	200
通过 JavaScript	201
Fade 淡入淡出	201
方法	201
.tab('dispose')	203
事件	203
导航栏(navbar)	204
运行原理	204
支持的内容	204
品牌	206
nav 导航	207
Form 表单	209
Text 文本处理	211
Color 颜色选择器(配色方案)	212
.Container 主内容-容器	212
定位	213
响应式行为处理	214
Toggler 切换触发器	214
扩展导航区内容	217
分页(Pagination)	218
概览	218

使用图标	218
禁用和活动状态定义	219
规格尺寸	220
对齐	221
POP 提示 (Popover)	223
概览	223
示例：在任意位置启用弹出窗口	223
示例：使用 container 容器选项	223
静态提示框	224
现场演示	224
四个方向	224
在下次点击时收回	225
禁用的元素	225
用法	226
选项	226
方法	227
Events 事件	229
进度条(Progress)	230
运行原理	230
标签	231
高度	231
背景	231
多进度条进度（嵌套）	232
条纹进度指示	232
动画条纹进度指示	233
滚动监听(Scrollspy)	234
运行原理	234
在 navbar 导航中的示例	234
嵌套的导航示例	235
列表组示例	236
用法	237
通过数据属性	237
通过 JavaScript	237
方法	238
.scrollspy('dispose')	238
选项	238
事件	238
提示冒泡(Tooltip)	239
概览	239
示例：在任何地方启用 tooltip 提示冒泡插件	239
实际范例	239
静态演示	239
互动演示	240
用法	240
标记	241
选项	241
方法	242
活动	244
公共样式	245

边框(border).....	245
基本边框	245
边框颜色	245
圆角边框	246
Clearfix 清动浮动.....	247
关闭图标	247
颜色	247
背景颜色	248
背景渐变	249
Display 显示属性	249
Display 运行原理	250
Dislay 属性.....	250
实例	250
隐藏的元素	251
面向打印的显示属性控制处理	251
嵌入	252
关于	252
示例	252
长宽比例处理	252
Flex 弹性布局.....	253
启用弹性行为	253
方向	253
内容对齐与对准	255
对齐项目	256
自对齐	257
自浮动 Auto margins	259
与 align-items 结合实现垂直布局.....	259
Wrap 包裹.....	260
Order 排序	261
对齐内容	263
Float 浮动属性	265
概览	265
Class 样式	265
图像替换	266
固顶(底)及定位.....	266
规格与尺寸(Sizing)	267
spacing 间隔规范（Margin 与 Padding 间距处理）	268
显示或隐藏(能见度)处理	270

简介

Bootstrap 是最受欢迎的前端框架，用于构建响应式、移动设备优先的网站。快速了解 Bootstrap 、使用 BootCDN 以及熟悉初学者模板页面。

快速开始

想要快速地将 Bootstrap 应用到你的项目中吗？推荐使用 Bootstrap 中文网 维护的 BootCDN 免费加速服务。是否还需要使用包管理工具或下载源码？[请进入下载页面。](#)

CSS

将引入 Bootstrap 样式表的 `<link>` 标签复制并粘贴到 `<head>` 中，并放在所有其他样式表之前。

复制

```
<link rel="stylesheet"
href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aowXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">
```

JS

Bootstrap 中的许多组件需要依赖 JavaScript 才能运行。具体来说，他们依赖的是 [jQuery](#)、[Popper.js](#) 以及我们自己开发的 JavaScript 插件。具体操作就是将下列 `<script>` 标签放到页面底部的 `</body>` 标签之前。注意顺序，jQuery 必须放在最前面，然后是 Popper.js，最后是我们自己的 JavaScript 插件。

我们使用的是 [jQuery's slim build](#)（即，[瘦身版](#)）版本，也同时支持完整版本。

复制

```
<script src="https://cdn.bootcss.com/jquery/3.2.1/jquery.slim.min.js"
integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KChRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN" crossorigin="anonymous"></script>
<script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q" crossorigin="anonymous"></script>
<script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8j0t6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl" crossorigin="anonymous"></script>
```

好奇哪些组件明确要求 jQuery、我们自己的 JS 和 Popper.js？请点击下面的链接查看。如果你不清楚一般结构如何组织的，请继续阅读下面的模板实例。

展示组件对 JavaScript 的依赖情况

初学者模板

务必用最新的设计和开发标准武装你的页面。这意味着使用 HTML5 doctype 声明、添加一个 `viewport` 标签让页面正确支持响应式布局。将这些整合在一起后，你的页面应当像下面这样：

复制

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6
JXm" crossorigin="anonymous">

    <title>Hello, world!</title>
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://cdn.bootcss.com/jquery/3.2.1/jquery.slim.min.js"
integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCKRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5
KkN" crossorigin="anonymous"></script>
    <script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"
integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0
b4Q" crossorigin="anonymous"></script>
    <script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVC
mY1" crossorigin="anonymous"></script>
  </body>
</html>
```

以上这些就是所有页面必须的。请访问 [布局](#) 或 [官方实例](#) 以作参考，然后就可以开始布局你的网站内容和组件了。

重要的全局样式和设置

Bootstrap 所使用的一些重要的全局样式和设置需要你在使用前务必了解，所有这些样式和设置都是以跨浏览器样式的**标准化**为目标的。下面让我们深入讲解。

HTML5 doctype

Bootstrap 要求设置 HTML5 doctype。如果没有这个设置，你会看到一些奇怪的、不完整的样式，但是只要添加了这个设置就不会出现任何错误了。

复制

```
<!doctype html>
<html lang="en">

...
</html>
```

响应式 meta 标签

Bootstrap 本着 *移动设备优先* 的策略开发的，按照这一策略，我们优先为移动设备优化代码，然后根据每个组件的情况并利用 CSS 媒体查询（CSS media queries）技术为组件设置合适的样式。为了确保在所有设备上能够正确渲染并支持触控缩放，务必将设置 **viewport** 属性的 **meta** 标签添加到 `<head>` 中。如下所示。

复制

```
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">
```

你可以在 [初学者模板](#) 中看到实例。

盒模型

为了在 CSS 中更直观的设置尺寸，我们将全局的 **box-sizing** 值从 **content-box** 修改为 **border-box**。这就确保了 **padding** 不会影响元素最终的宽度计算，但是这可能会导致第三方组件出现问题，例如 Google 地图和 Google 定制搜索。

在极少数的情况下你需要重置 **box-sizing** 的值，请参考如下代码：

复制

```
.selector-for-some-widget {
  box-sizing: content-box;
}
```

通过以上代码片段，嵌套元素（包括通过 `::before` 和 `::after` 生成的内容）都将继承 **.selector-for-some-widget** 指定的 **box-sizing** 值。

在 CSS Tricks 网站可以了解到更多关于 [盒模型及尺寸设置](#) 相关的知识。

Reboot

为了改善跨浏览器的渲染，我们使用 [Reboot](#) 修正跨浏览器和设备之间的不一致性，同时对常用的 HTML 元素设置统一的效果。

JavaScript 脚本

Bootstrap 包括一些 JavaScript 帮助用户做出更加充满生机、活动的项目。欢迎学习更多关于如何去调用动态事件、灵活展示的数据和编程性的 API 选项等。

单个引用或者编译

一个页面只要使用一个 Bootstrap 的单个 ***.js** 文件，一次性使用 **bootstrap.js** 或压缩精简的 **bootstrap.min.js**（不要两个都调用），单个 JS 插件可以被包含调用之。

依赖关系

一些插件和 CSS 组件依赖于其它插件，这在系统中是被允许的。如果你要单个调用插件，请先在文档中检查它们的依赖关系。还要注意，所有插件都依赖于 **jQuery**（即 **jQuery** 必须在所有 JS 插件文件之前调用）。参考 [Consult our package.json](#) 可以了解哪些详尽版本的 jQuery 是被支持的。

比如下拉菜单 **dropdowns**、提示组件 **popovers**、冒泡组件等都提依赖于 [Popper.js](#)。

数据属性

通过 HTML 的数据属性几乎能启用并配置所有的 Bootstrap 插件（我们更倾向于函数化地使用 JavaScript）。，住只能在单个元素上使用一套数据属性（例如，不能在一个按钮上同时触发一个工具提示和一个模态框）。

在某些情况下，可能需要禁用此功能。要禁用数据属性 API，可用下面的方法在文档中解绑所有的带 `data-api` 命名空间的事件：

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

如需要指向特定的插件，只需要在调用插件的全名空间上拼接 `data-api` 作为命名空间，如下所示：

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

JS 事件(Event)

Bootstrap 为大多数插件的独一无二的行为提供了自定义事件。通常情况下，这里会有动词不定式和过去分词形式—如果在事件的开始触发了它的动词不定式（如 `show`），而 `shown` 在完成某个动作后触发其过去分词形式（如 `shown`）。

所有不定式事件都提供 `preventDefault()` 功能，从而使开发者设计一个动作开始之前就能终止它的执行。从事件处理程序返回 `false` 同样也是自动调用 `preventDefault()` 功能。

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

编程化的 API

我们还相信，你能够纯粹通过 JavaScript API 来使用所有的 Bootstrap 插件。所有的公共的 API 都是单一的，可链接的方法，并返回执行的集合。

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

所有的方法都能够接收一个可取舍的 `options` 对象、一个指向特定方法的字符串，或者不接收参数（不带参数地调用这个方法将用默认行为初始化一个插件）：

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

每个插件都需要在 `Constructor` 属性上明文曝露它的原始构造函数 `$.fn.popover.Constructor`。如果你想获得一个特定的插件实例，可以从一个元素中直接获得它： `$('[rel="popover"]').data('popover')`。

异步函数和转换

All programmatic API methods are **asynchronous** and returns to the caller once the transition is started **but before it ends**.

所有编程化的 API 方法均为异步，一旦转换开始并在结束之前返回给调用者。如果需要在转换完成后执行动作，您可以收听相应的事件（进行侦听并作出下一步的编程）：

```
$('#myCollapse').on('shown.bs.collapse', function (e) {  
  // Action to execute once the collapsible area is expanded  
})
```

另外，**transition** 过渡组件上的方法调用将被忽略：

```
$('#myCarousel').on('slid.bs.carousel', function (e) {  
  $('#myCarousel').carousel('2') // Will slide to the slide 2 as soon as the  
  transition to slide 1 is finished  
})
```

```
$('#myCarousel').carousel('1') // Will start sliding to the slide 1 and returns to  
the caller
```

```
$('#myCarousel').carousel('2') // !! Will be ignored, as the transition to the slide 1 is not finished !!
```

默认设置

用户可以通过修改插件的 `Constructor.Default` 对象来更改插件的默认设置：

```
$.fn.modal.Constructor.Default.keyboard = false // changes default for the modal plugin's `keyboard` option to false
```

无冲突处理

有时，必须使用 Bootstrap 插件和其他 UI 框架。在这种情况下，偶尔会发生命名空间冲突。如果发生这种情况，您可以调用 `.noConflict` 恢复插件的值：

```
var bootstrapButton = $.fn.button.noConflict() // return $.fn.button to previously assigned value
$.fn.bootstrapBtn = bootstrapButton           // give $.fn.bootstrapBtn the Bootstrap functionality
```

版本管理

每个 Bootstrap 的 jQuery 插件的版本都可以通过 `VERSION` 插件的构造函数的属性访问。如对于 `tooltip` 工具提示插件：
`$.fn.tooltip.Constructor.VERSION // => "4.0.0-beta.2"`

当 JavaScript 被禁用时，没有特殊的回调机制

Bootstrap 插件在 JavaScript 被禁用时没有特殊的回调方式。如果你比较关心用户体验的话，可用 `<noscript>` 向你的用户解释情况（以及指引如何重新开启 JavaScript），然后/或者添加自定义回调机制。

第三方库

Bootstrap 并不正式支持第三方的 JavaScript 库，比如 Prototype 或者 jQuery UI。尽管有系统中 `.noConflict` 和命名空间事件，但如果混合引用可能会带来兼容性问题（高手不怕脏和累、并自行修复 BUG 条件下，请自便:）。

Util 方法

默认 `bootstrap.js`（预编译与精简版）都已经包含了 `util.js`，因为 Bootstrap 所有 JavaScript 行为都依赖于 `util.js` 函数。

`util.js` 包括效用函数和 `transitionEnd` 事件的基本帮助器以及 CSS 转换仿真器。它被其他插件用于检查 CSS 过渡支持并捕获挂起的过渡。

Bootstrap 包含了一些供你的项目使用的组件和选项，包括外包裹容器、强大的网格系统、灵活多变的媒体对象和响应式的工具类。

Container 容器

Container 容器是窗口布局的最基本元素，我们推荐所有样式都定义在 `.container` 或 `.container-fluid` 容器之中--**这是启用整个栅格系统必不可少的前置条件**，它们分别对应选择一个响应式的、固定宽度的容器，或者选择一个流式自适应浏览器或容器最大合法宽度的窗口（意味着任何时候它的宽度总是 100%）。

`.container` 容器 *可以被嵌套*，但是大多数布局并不需要这么做（最少层次的嵌套构建出的网页更优雅-译者注），其呈现效果和使用方法如下所示：



```
<div class="container">
  <!-- Content here -->
</div>
```

使用 `.container-fluid` 类，可以使 `div` 宽度扩展到整个宽度（如果没有被其它 CSS 容器包含，则应是浏览器运行时的宽度，否则应是父容器中允许的最大宽度，一般视为 100% 宽度），示例效果和代码使用方法如下：



```
<div class="container-fluid">
  ...
</div>
```

响应式的分界点

Bootstrap 是基于移动优先的原则开发的，使用了一系列的媒体查询（[media queries](#)）方法，为我们的布局和界面创建自适应的分界点。这些分界点主要是基于视口宽度的最小值，并且当窗口视图改变的时候允许元素缩放。

在 Bootstrap 的源 Sass 文件中，为了实现布局、网格系统以及组件，首先使用下面的媒体查询范围（可以理解为将不同宽度的网页进行拆分并分别载入 CSS 样式处理构建）：

```
// Extra small devices (portrait phones, less than 576px)
// No media query since this is the default in Bootstrap
// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) { ... }

// Medium devices (tablets, 768px and up)
@media (min-width: 768px) { ... }

// Large devices (desktops, 992px and up)
@media (min-width: 992px) { ... }

// Extra large devices (large desktops, 1200px and up)
@media (min-width: 1200px) { ... }
```

由于我们在 Sass 中写了源 CSS，所有的媒体查询通过 Sass mixins 都是可用的（SASS 的 Mixins 可以一次性定义功能模块，让你在任何地方调用，并且可以无限制的重用）：

```
@include media-breakpoint-up(xs) { ... }
@include media-breakpoint-up(sm) { ... }
@include media-breakpoint-up(md) { ... }
@include media-breakpoint-up(lg) { ... }
@include media-breakpoint-up(xl) { ... }
```

// Example usage:

```
@include media-breakpoint-up(sm) {
  .some-class {
    display: block;
  }
}
```

偶尔也会使用其它方面的媒体查询（指定屏幕的尺寸或更小）：

```
// Extra small devices (portrait phones, less than 576px)
@media (max-width: 575px) { ... }
```

```
// Small devices (landscape phones, less than 768px)
@media (max-width: 767px) { ... }
```

```
// Medium devices (tablets, less than 992px)
@media (max-width: 991px) { ... }
```

```
// Large devices (desktops, less than 1200px)
@media (max-width: 1199px) { ... }
```

// Extra large devices (large desktops)

// No media query since the extra-large breakpoint has no upper bound on its width

请注意，由于浏览器目前不支持 [范围方面的查询](#)，我们解决的局限性 [min- 和 max- 前缀](#) 和视口带小数的宽度（可下的高 DPI 设备一定的条件下发生，例如）通过使用值与这些比较高的精度。

同样，这些媒体查询通过 Sass mixins 也是可用的：

```
@include media-breakpoint-down(xs) { ... }
@include media-breakpoint-down(sm) { ... }
@include media-breakpoint-down(md) { ... }
@include media-breakpoint-down(lg) { ... }
```

以及使用最小和最大断点宽度定位单个屏幕尺寸段的媒体查询和混合定义：

```
// Extra small devices (portrait phones, less than 576px)
@media (max-width: 575px) { ... }
```

```
// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) and (max-width: 767px) { ... }
```

```
// Medium devices (tablets, 768px and up)
@media (min-width: 768px) and (max-width: 991px) { ... }
```

```
// Large devices (desktops, 992px and up)
@media (min-width: 992px) and (max-width: 1199px) { ... }
```



```
// Extra large devices (large desktops, 1200px and up)
```

```
@media (min-width: 1200px) { ... }
```

这样媒体查询同理可通过 Sass mixins 获得：

```
@include media-breakpoint-only(xs) { ... }
```

```
@include media-breakpoint-only(sm) { ... }
```

```
@include media-breakpoint-only(md) { ... }
```

```
@include media-breakpoint-only(lg) { ... }
```

```
@include media-breakpoint-only(xl) { ... }
```

换一个维度看问题，媒体查询可以跨越多个断点宽度：

```
// Example
```

```
// Apply styles starting from medium devices and up to extra large devices
```

```
@media (min-width: 768px) and (max-width: 1199px) { ... }
```

针对同一屏幕尺寸范围的 Sass mixin 将是如下定义方法：

```
@include media-breakpoint-between(md, xl) { ... }
```

Z-index 堆叠顺序属性

若干个 Bootstrap 组件利用 z-index CSS 属性，通过提供第三轴来安排内容来帮助控制布局。我们在 Bootstrap 中使用默认的 z-index 量表，该缩放比例设计用于正确地分层导航、工具提示、扩展插件、模态框等场合。

一般情况下，不推荐用户去自定义这些属性，否则可能牵一发而动全，影响全响呈现。以下为 Bootstrap 系统内置的 Z-index 堆叠顺序属性清单：

```
$zindex-dropdown: 1000 !default;
```

```
$zindex-sticky: 1020 !default;
```

```
$zindex-fixed: 1030 !default;
```

```
$zindex-modal-backdrop: 1040 !default;
```

```
$zindex-modal: 1050 !default;
```

```
$zindex-popover: 1060 !default;
```

```
$zindex-tooltip: 1070 !default;
```

Background 背景元素会忽略 z-index 属性定义，而普通背景倾向于驻留在较低的位置，而导航和移动使用更高的层次 z-index 来确保导航层能覆盖其它控件，实现浮动、导航等设计效果。

栅格系统

Bootstrap 包含了一个强大的移动优先的网格系统，它是基于一个 12 列的布局、有 5 种响应尺寸(对应不同的屏幕)，支持 Sass mixins 自由调用，并结合自己预定义的 CSS、Js 类，用来创建各种形状和尺寸的布局。

机制原理

Bootstrap 的网格系统使用一系列 div 容器的行、列来布局和对齐内容，不同于旧版 3.0，新版是完全基于 flexbox 流式布局构建的，完全支持响应式标准。下面的示例，可以让我们深入了解网格如何组合在一起。

新手不熟悉 flexbox？ 阅读这篇关于 flexbox 布局文章，可以了解其背景、术语、规则和示例代码。

中文版的 FlexBox 布局教程可见：<https://www.z01.com/help/web/3234.shtml>（译者注）

三分之一空间占位

三分之一空间占位

三分之一空间占位


```
<div class="container">
  <div class="row">
    <div class="col-sm">
      三分之一空间占位
    </div>
    <div class="col-sm">
      三分之一空间占位
    </div>
    <div class="col-sm">
      三分之一空间占位
    </div>
  </div>
</div>
```

上面的例子使用 Bootstrap 预定义的栅格系统，演示了在 `.container` 容器内建立了三个等宽的列，并且分别兼容在 `small`(极窄宽度网页)、`medium`(中等宽度网页)、`large`(宽网页)、`extra large`(超宽网页)四种设备类型-即无论网页宽度如何，这三个列都是**恒在**呈现的。

让我们来慢慢揭开它的工作原理：

- 栅格系统提供了集中内容居中、水平填充网页内容的方法，使用 `.container`(严格意义上也包括 `.container-fluid`，后文相同不再备注-译者)应答网页宽度，或使用 `.container-fluid` 使网页能够以 **100%**宽度呈现在所有的浏览器窗口或设备尺寸上。
换一个说法就是：`.container` 实现固定的宽度并居中呈现，`.container-fluid` 实现全宽度，并和其它网格实现对齐(译者注)。
- 行(`.row`)是列(`.col-*`)的横向组合和父容器(它们有效组织在 `.row` 下)，每列都有水平的 `padding` 值，用于控制它们之间的间隔，同时在负边距的行上抵消，从而实现列中的所有内容在视觉上是左侧对齐的体验。
- 网页开发者的呈现内容必须放置在列(`.col-*`)中，而且只有列可以是行的直接子元素，否则都是违法的(不可以在 `.col-*` 以上添加呈现内容)。
- 这一切都要感谢 `flexbox` 流式布局，从而使我们不需要指定列的宽度(旧版 Bootstrap3 是采用严格宽度定义来实现的)就能实现网页自动等宽排列，比如我们在 `.container` 中置入初始化的四个 `.col-sm` 就能实现各自 25%宽度并左对齐形成一行的排列。更多示例，请参阅本文档 [自动布局列](#) 部分。
- 你可能注意到 `.col-*` 后面有不同的数字，如 `.col-sm-4` 或 `.col-xl-12`，这些 `css` 类后面的数字用于表明定义 `div` 空间想要占用列的数量，每行最多有 12 列。如果你想用三个等宽的列，则取 12 的三分之一，即 `.col-sm-4` 就是正确的(后文会有详细的介绍)。
- `.col-*` 的 `width` 属性(即列宽)是用百分比来表现和定义的，所以它们总是流式的，其尺寸大小受父元素的定义影响(这正是 `flexbox` 布局的特征，子元素的宽比和排列受父元素定义)。
- 列具有水平 `padding` 定义，用于创建列与列之间的间隙。
- `.row` 上带有 `margin-left: -15px;margin-right: -15px;` 属性，你可以在 `.row` 上定义 `.no-gutters` 属性，从而消除这个属性，使页面不会**额外宽出 30px**，即 `<div class="row no-gutters">...`。(译者原意拆成两行表述)。
- 总共有**五个栅格等级**，每个响应式分界点隔出一个等级：特小 `.col`、小 `.col-sm-*`、中 `.col-md-*`、大 `.col-lg-*`、特大(大、特大也可以称为宽、超宽) `.col-xl-*`。
- 栅格断点的媒体查询基于宽度的最小值，意味着它们应用到某一等级以及这一等级之上的所有(如 `.col-sm-4` 的定义可以在小型、中型、宽、超宽设备上呈现，但不适用于能在超小型 `.col-sx` 上呈现)。
- 用户不需要自行定义网格，可以直接使用系统预定义的栅格类(如 `.col-4`)或采用 `Sass mixins` 来进行更多的语义标记满足开发需要。

请注意：Flexbox 布局虽然很先进，但也有一定的**限制和错误**，如无法使用某些 HTML 元素作为弹性容器，[点此可扩展相关知识](#)。

栅格选项

Bootstrap 使用 **ems** 或 **rems** 来定义大多数属性的规格大小、**px** 用于全局层面网格断点和容器宽度（因为浏览器和设备的宽度是以像素 **px** 为单位，且不会随字体大小而变化）。

通过一个简单的表格查看 Bootstrap 的网格系统在各种屏幕和设备上的细节约定：

	超小屏幕 (新增规格)<576px	小屏幕 次小屏≥576px	中等屏幕 窄屏≥768px	大屏幕 桌面显示器≥992px	超大屏幕 大桌面显示器 ≥1200px
.container 最大宽度	None (auto)	540px	720px	960px	1140px
类前缀	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-
列（ column ）数	12				
列间隙	30px (每列两侧各 15px)				
可嵌套性	Yes				
可排序性	Yes				

译者注：

- 1、在 Bootstrap 4 中，屏幕的大小是真正的“断点”，即如果只定义一个屏幕规格即可向上覆盖所有设备，向下如果没有定义则默认为 12 栅格占位-译者注。
- 2、有版本将 540px 断点翻译为平板，目前业界平板(如 ipad)进入高清屏幕时代，故本翻译不作此处理以符发展需求。
- 3、实践中，一个<div class="col"></div>或<div class="col-N"></div>代表手机断点

自动布局列

利用栅格断点特性进行排版，可以简化列的大小，而不需要批定显式的列宽，如强制写为：**.col-sm-6**。

等宽布局

下面的列子，展示了一行两列与一行三列的布局，从 **xs**（如上表如见，实际上并不存在 **xs** 这个空间命名，是以 **.col** 表示，下同不再注）到 **x1**（即 **.col-xl-***）所有设备上都是等宽并占满一行，只要简单的应用 **.col** 就可以完成。

1 of 2	2 of 2	
1 of 3	2 of 3	3 of 3

```
<div class="container">
  <div class="row">
    <div class="col">
      1 of 2
    </div>
    <div class="col">
```

```

    2 of 2
  </div>
</div>
<div class="row">
  <div class="col">
    1 of 3
  </div>
  <div class="col">
    2 of 3
  </div>
  <div class="col">
    3 of 3
  </div>
</div>
</div>

```

等宽列可以分成多行，但是有一个 [Safari 旧版浏览器的 flexbox 错误](#)，阻止它在没有显示 `flex-basis` 或 `border`（影响到边框效果）。

Bootstrap 非官方的一个实例，已经介绍了[两种解决方法](#)，但如果是最新 safari 浏览器下，则不需要这样做。

下面是等宽列两行的处理方法，引用 `w-100` 进行切割分行：

Column	Column
Column	Column

```

<div class="container">
  <div class="row">
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="w-100"></div>
    <div class="col">Column</div>
    <div class="col">Column</div>
  </div>
</div>

```

设置一列宽度

在 Flexbox 的布局上，拥有很多现代特征，比如自动布局和列宽处理。你可以在一行多列的情况下，特别指定一列并进行宽度定义，同时其它列自动调整大小，可以使用预定义的栅格类（如下所示），从而实行栅格宽或行宽的优化处理。注意在这种民培上，无论中心定义列的宽度如何，其他列都将调整大小。

1 of 3	2 of 3 (更宽-12格中占6格，其它6格另外两列平分)	3 of 3
1 of 3	2 of 3 (更宽-12格中占5格,其它7格另外两列平分-不论奇偶都能达成)	3 of 3

```
<div class="container">
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col-6">
      2 of 3 (更宽-12 格中占 6 格，其它 6 格另外两列平分)
    </div>
    <div class="col">
      3 of 3
    </div>
  </div>
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col-5">
      2 of 3 (更宽-12 格中占 5 格,其它 7 格另外两列平分-不论奇偶都能达成)
    </div>
    <div class="col">
      3 of 3
    </div>
  </div>
</div>
```

可变宽度的弹性空间

使用 `col-{breakpoint}-auto` 断点方法，可以实现根据其内容的自然宽度来对列进行大小调整。。

1 of 3	可变宽度内容自由伸张，左右宽度不变。	3 of 3
1 of 3	可变宽度内容自由伸张,左列宽度变化(右列绑定col-lg-2栅格数)	3 of 3

```
<div class="container">
  <div class="row justify-content-md-center">
    <div class="col col-lg-2">
      1 of 3
    </div>
    <div class="col-md-auto">
```

```

        Variable width content
    </div>
    <div class="col col-lg-2">
        3 of 3
    </div>
</div>
<div class="row">
    <div class="col">
        1 of 3
    </div>
    <div class="col-md-auto">
        Variable width content
    </div>
    <div class="col col-lg-2">
        3 of 3
    </div>
</div>
</div>

```

等宽多行

创建跨多个行的等宽列，方法是插入.w-100 要将列拆分为新行。通过混合.w-100 一些还可以影响一些显示状态效果,如按钮排序等。

译者注：这部份其实和上面 safri 浏览器 bug 讲解重复了，但官网如此保持尊重。

又注：.w100 似乎与.clearfix 有时可以达到同样的网页效果。

col	col
col	col

```

<div class="row">
    <div class="col">col</div>
    <div class="col">col</div>
    <div class="w-100"></div>
    <div class="col">col</div>
    <div class="col">col</div>
</div>

```

响应式的 class 选择器

Bootstrap 的栅格系统包括五种宽带预定义，用于构建复杂的响应布局，你可以根据需要定义在特小.col、小.col-sm-*、中.col-md-*、大.col-lg-*、特大.col-xl-*五种屏幕(设备)下的样式。

覆盖所有设备

如果要一次性定义从最小设备到最大设备相同的网格系统布局表现，请使用.col和.col-*类。后者是用于指定特定大小的(如.col-6)，否则使用.col就可以了。

col	col	col	col
col-8			col-4

```
<div class="row">
  <div class="col">col</div>
  <div class="col">col</div>
  <div class="col">col</div>
  <div class="col">col</div>
</div>
<div class="row">
  <div class="col-8">col-8</div>
  <div class="col-4">col-4</div>
</div>
```

水平堆砌

使用单一的.col-sm-*类方法，可以创建一个基本的网格系统，此时如果没有指定其它媒体查询断点宽度，这个栅格系统是成立的，而且会随着屏幕变窄成为超小屏幕.col-后，自动成为每列一行、水平堆砌。改变网页屏幕宽度你可以在下面列子看到效果：

col-sm-8		col-sm-4
col-sm	col-sm	col-sm

```
<div class="row">
  <div class="col-sm-8">col-sm-8</div>
  <div class="col-sm-4">col-sm-4</div>
</div>
<div class="row">
  <div class="col-sm">col-sm</div>
  <div class="col-sm">col-sm</div>
  <div class="col-sm">col-sm</div>
</div>
```

混合布局

设计师们当然不会简单的把各个屏幕下的栅格都做成一样，那将是单调乏味的，所以你可以根据需要对每一个列进行不同的设备定义。下方示例展示了原理：

.col-12 .col-md-8		.col-6 .col-md-4
.col-6 .col-md-4	.col-6 .col-md-4	.col-6 .col-md-4
.col-6		.col-6

<!-- 定义在超小屏幕下 1 列全宽、1 列半宽，而其它场景以 8:4 比例并行排列 -->

```
<div class="row">
  <div class="col-12 col-md-8">.col-12 .col-md-8</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>
```

<!-- Columns start at 50% wide on mobile and bump up to 33.3% wide on desktop -->

```
<div class="row">
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>
```

<!-- Columns are always 50% wide, on mobile and desktop -->

```
<div class="row">
  <div class="col-6">.col-6</div>
  <div class="col-6">.col-6</div>
</div>
```

对齐

flexbox 布局可以轻松的实现 DIV 空间布局的垂直、水平对齐。

垂直对齐示例

上边贴齐	上边贴齐	上边贴齐s
上下居中对齐	上下居中对齐	上下居中对齐
下边对齐	下边对齐	下边对齐

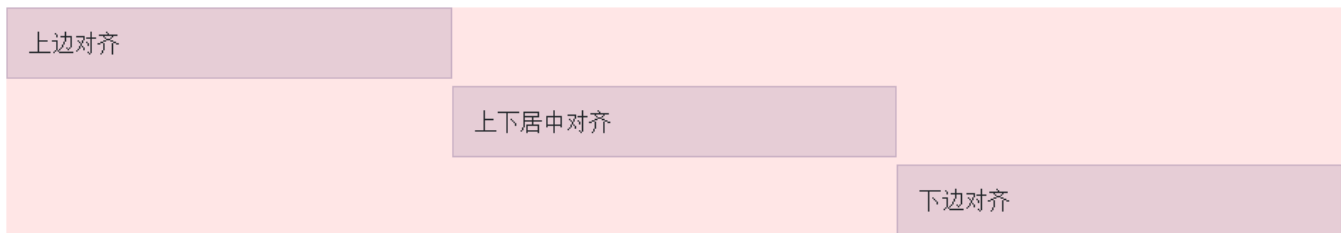
```

<div class="container">
  <div class="row align-items-start">
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
  </div>
  <div class="row align-items-center">
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
  </div>
  <div class="row align-items-end">
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
  </div>

```



```
</div>
<div class="col">
  One of three columns
</div>
</div>
</div>
```



```
<div class="container">
  <div class="row">
    <div class="col align-self-start">
      One of three columns
    </div>
    <div class="col align-self-center">
      One of three columns
    </div>
    <div class="col align-self-end">
      One of three columns
    </div>
  </div>
</div>
```

水平对齐示例



```
<div class="container">
  <div class="row justify-content-start">
    <div class="col-4">
      One of two columns
    </div>
```

```

    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-center">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-end">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-around">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-between">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
</div>

```

间隙沟槽(gutters)清除

Bootstrap 默认的栅格和列间有间隙沟槽，一般是左右-15px 的 `margin` 或 `padding` 处理，您可以使用 `.no-gutters` 类来消除它，这将影响到 `.row` 行、列平行间隙及所有子列。

以下是创建这些样式的源代码。注意，列替换仅限于第一个子列，并通过属性选择器进行定位。当这产生一个更具体的选择器时，列填充仍然可以使用间隔实用程序进一步定制。

如果你需要**无边缝设计(edge-to-edge design)**，则请在父 DIV 中放弃 `.container` 与 `.container-fluid` 容器。

```
.no-gutters {
  margin-right: 0;
  margin-left: 0;

  > .col,
  > [class*="col-"] {
    padding-right: 0;
    padding-left: 0;
  }
}
```

下方为实际效果展示，您可以继续使用所有 Bootstrap 预定义的栅格系统（包括列宽、响应层、属性、排序等）

```
.col-12 .col-sm-6 .col-md-8
```

```
.col-6 .col-md-4
```

```
<div class="row no-gutters">
  <div class="col-12 col-sm-6 col-md-8">.col-12 .col-sm-6 .col-md-8</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>
```

列换行

如果在一行内子 DIV 定义的栅格总数超过 12 列，Bootstrap 会在保留列完整的前提下，将无法平行布局的多余列，重置到下一行，并占用一个完整的新行。

```
.col-9
```

```
.col-4
```

Since $9 + 4 = 13 > 12$, this 4-column-wide div gets wrapped onto a new line as one contiguous unit.

```
.col-6
```

Subsequent columns continue along the new line.

```
<div class="row">
  <div class="col-9">.col-9</div>
  <div class="col-4">.col-4<br>Since  $9 + 4 = 13 > 12$ , this 4-column-wide div gets
wrapped onto a new line as one contiguous unit.</div>
  <div class="col-6">.col-6<br>Subsequent columns continue along the new
line.</div>
</div>
```

换行

一般换行推荐使用添加多个 `.row` 来达成，否则你可以使用系统提供的 `.w-100` 方法处理，其思路是强行插入一个 `width:100%` 的 DIV 进行隔离切断（前文有两处提到，这是 Flexbox 流式布局的一个 Hack，目前没有更好的方案）。

.col-6 .col-sm-3	.col-6 .col-sm-3
.col-6 .col-sm-3	.col-6 .col-sm-3

```
<div class="row">
  <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>
  <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

  <!-- Force next columns to break to new line -->
  <div class="w-100"></div>

  <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>
  <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>
</div>
```

也可以结合[响应式实用程序](#)来换行(切割)。

.col-6 .col-sm-4	.col-6 .col-sm-4
.col-6 .col-sm-4	.col-6 .col-sm-4

```
<div class="row">
  <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>
  <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>

  <!-- Force next columns to break to new line at md breakpoint and up -->
  <div class="w-100 d-none d-md-block"></div>

  <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>
  <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>
</div>
```

重排序

Class 顺序重定义

使用 `.order-*` class 选择符，可以对 DIV 空间进行 **可视化排序**，系统提供了 `.order-1` 到 `.order-12` 12 个级别的顺序，在五种浏览器和设备宽度上都能生效。

1号空间-未定义序号，位置不变。	3号空间-放第1但受1号空间不变影响居第2位。	2号空间-排最后。
------------------	-------------------------	-----------

```

<div class="container">
  <div class="row">
    <div class="col">
      1 号空间-未定义序号，位置不变。
    </div>
    <div class="col order-12">
      2 号空间-排最后。
    </div>
    <div class="col order-1">
      3 号空间-放第 1 但受 1 号空间不变影响居第 2 位。
    </div>
  </div>
</div>

```

还可以使用 `.order-first`，快速更改一个顺序到最前面，同时其它元素也相应的获得了 `order:-1` 的属性，这个属性也可以与 `.order-*` 混合使用。

```

<div class="container">
  <div class="row">
    <div class="col">
      1 号空间-未定义顺序。
    </div>
    <div class="col">
      2 号空间-未定义顺序。
    </div>
    <div class="col order-first">
      3 号空间-优先排序，占第 1 位。
    </div>
  </div>
</div>

```

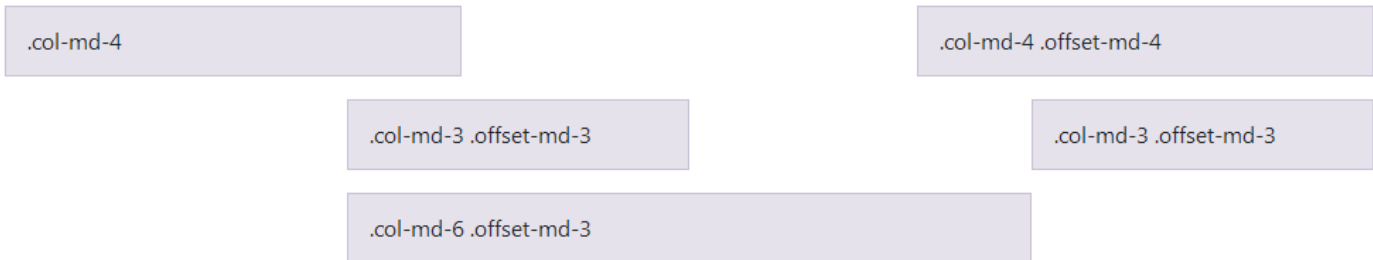
列偏移

可以使用两种方式进行列偏应：

- 1、使用响应式的 `.offset-*` 栅格偏移方法。
- 2、使用[边界处理实用程序](#)，它内置了诸如 `.ml-*`、`.p-*`、`.pt-*` 等实用排工具。

class 偏移选择器

使用 `.offset-md-*` 类可以使列向右偏移，通过定义 `*` 的数字，则可以实现列偏移，如 `.offset-md-4` 则是向右偏移四列。



```

<div class="row">

```

```

<div class="col-md-4">.col-md-4</div>
<div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
</div>
<div class="row">
  <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
  <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
</div>
<div class="row">
  <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>
</div>

```

除了在响应断点处清除列，您可能需要重置偏移量，下面的[栅格示例展示](#)了这一点：

.col-sm-5.col-md-6	.col-sm-5.offset-sm-2.col-md-6.offset-md-0
.col.col-sm-6.col-md-5.col-lg-6	.col-sm-6.col-md-5.offset-md-2.col-lg-6.offset-lg-0

```

<div class="row">
  <div class="col-sm-5 col-md-6">.col-sm-5 .col-md-6</div>
  <div class="col-sm-5 offset-sm-2 col-md-6 offset-md-0">.col-sm-5 .offset-sm-2 .col-md-6 .offset-md-0</div>
</div>

<div class="row">
  <div class="col-sm-6 col-md-5 col-lg-6">.col.col-sm-6.col-md-5.col-lg-6</div>
  <div class="col-sm-6 col-md-5 offset-md-2 col-lg-6 offset-lg-0">.col-sm-6 .col-md-5 .offset-md-2 .col-lg-6 .offset-lg-0</div>
</div>

```

Margin 移动布局

在 Bootstrap V4 中，您可以使用 `.ml-auto` 与 `.mr-auto` 来强制隔离两边的距离，实现类水平隔离的效果。

.col-md-4	.col-md-4.ml-auto
.col-md-3.ml-md-auto	.col-md-3.ml-md-auto
.col-auto.mr-auto	.col-auto

```

<div class="row">
  <div class="col-md-4">.col-md-4</div>
  <div class="col-md-4 ml-auto">.col-md-4 .ml-auto</div>
</div>
<div class="row">
  <div class="col-md-3 ml-md-auto">.col-md-3 .ml-md-auto</div>
  <div class="col-md-3 ml-md-auto">.col-md-3 .ml-md-auto</div>
</div>

```

```
<div class="row">
  <div class="col-auto mr-auto">.col-auto .mr-auto</div>
  <div class="col-auto">.col-auto</div>
</div>
```

列嵌套

为了使用内置的栅格系统将内容再次嵌套，可以通过添加一个新的 `.row` 元素和一系列 `.col-sm-*` 元素到已经存在的 `.col-sm-*` 元素内。被嵌套的行（row）所包含的列（column）数量推荐不要超过 12 个（其实，没有要求你必须占满 12 列-否则应对页面进行重新规划布局）。



```
<div class="row">
  <div class="col-sm-9">
    Level 1: .col-sm-9
    <div class="row">
      <div class="col-8 col-sm-6">
        Level 2: .col-8 .col-sm-6
      </div>
      <div class="col-4 col-sm-6">
        Level 2: .col-4 .col-sm-6
      </div>
    </div>
  </div>
</div>
```

Sass mixins

用 Bootstrap 的源 Sass 文件时，你可以选择使用 Sass 变量，或者 mixins，以创建自定义的、语义化的、响应式的网页布局。我们的预建类使用相同的变量以及 mixins，为快速响应布局提供整个现成的套件。

变量

变量决定列的数量、缝隙宽度以及切换到浮动列的媒体查询点，们用这些来生成上文提及的预定义网格列，定制出如下所示的自定义 mixins:

```
$grid-columns: 12;
$grid-gutter-width: 30px;

$grid-breakpoints: (
  // Extra small screen / phone
  xs: 0,
  // Small screen / phone
```

```

    sm: 576px,
    // Medium screen / tablet
    md: 768px,
    // Large screen / desktop
    lg: 992px,
    // Extra large screen / wide desktop
    xl: 1200px
  );

$container-max-widths: (
  sm: 540px,
  md: 720px,
  lg: 960px,
  xl: 1140px
);

```

Mixins

MMixins 用来联合网格变量，为每个网格列生成语义化的 CSS。

```

// Creates a wrapper for a series of columns
@include make-row();

// Make the element grid-ready (applying everything but the width)
@include make-col-ready();
@include make-col($size, $columns: $grid-columns);

// Get fancy by offsetting, or changing the sort order
@include make-col-offset($size, $columns: $grid-columns);

```

应用案例

你可以将变量修改为你自己定义的值，或者直接使用 mixins 的默认值。下面例子即使用默认设置创建两列布局以及一个列与列之间的缝隙：

```

.example-container {
  width: 800px;
  @include make-container();
}

.example-row {
  @include make-row();
}

.example-content-main {
  @include make-col-ready();

  @include media-breakpoint-up(sm) {

```



```

    @include make-col(6);
  }
  @include media-breakpoint-up(lg) {
    @include make-col(8);
  }
}

```

```

.example-content-secondary {
  @include make-col-ready();

  @include media-breakpoint-up(sm) {
    @include make-col(6);
  }
  @include media-breakpoint-up(lg) {
    @include make-col(4);
  }
}

```

Main content

Secondary content

```

<div class="example-container">
  <div class="example-row">
    <div class="example-content-main">Main content</div>
    <div class="example-content-secondary">Secondary content</div>
  </div>
</div>

```

自定义栅格系统

使用 Bootstrap v4 内置的栅格系统之 Sass 变量和 maps 文件，可以完全自定义自己的栅格格环境、并可以更改 DIV 层次、媒体查询维度和 container 容器宽度，然后重新编译。

自定义列和间隙

以通过修改 Sass 变量、重定义栅格系统列的数量，使用 `$grid-columns` 命令可生成每个单独列的宽度（以百分比表示），同时 `$grid-gutter-width` 允许在平行排列 `$padding-left` 和 `$padding-right` 对于平台间隔均匀分割的断点特定宽度。

```

$grid-columns: 12 !default;
$grid-gutter-width: 30px !default;

```

自定义栅格

超越自我，您还可以自定义网格层数。如果你只想要四个格子层，只要更新 `$grid-breakpoints` 和 `$container-max-widths` 等类参数。

```

$grid-breakpoints: (
  xs: 0,
  sm: 480px,

```

```
md: 768px,  
lg: 1024px  
);
```

```
$container-max-widths: (  
  sm: 420px,  
  md: 720px,  
  lg: 960px  
);
```

对 Sass 变量或 map 文件进行任何更改，都需要保存更改并重新编译。这样做将为列宽度，偏移量和排序输出一组全新的预定义栅格类。响应可见性实用程序也将更新为使用自定义断点，同时濂晨设置网格值时是采用 **px** 单位（而不是 **rem**、**em** 或 **%**）。

初始化与 CSS 重置

Bootstrap 致力于提供一个简洁、优雅的基础，以此作为立足点。我们使用 Reboot，把一系列元素特征的 CSS 修改放在一个文件里。

路线方针

系统重置建立新的规范化，只允许元素选择器向各个 HTML 元素提供了自有的风格，额外的样式只通过明确的 `.class` 类来规范。例如，我们重置了一系列 `<table>` 样式作为简单的基准，然后提供了 `.table`、`.table-bordered` 等样式类，从而最小限定的定义、最大程序兼容、最多场景可被引用。

以下是我们在重置 CSS 中选择覆盖和重定义哪些元素的指导方针和理由：

重置浏览器默认值，使用 `rem` 作为尺寸规格单位，代替 `em` 用于指定可缩放的组件的间隔与缝隙。

最大化避免使用 `margin-top`，防止使用它造成的垂直外排版（边距）混乱所造成之意想不到结果。更重要的是，一个单一方向的 `margin` 是一个简单的构思模型。

为了易于跨设备缩放，`block` 块元素必须使用 `rem` 作为 `margin` 的单位。

保持 `font` 相关属性最小的声明，尽可能地使用 `inherit` 属性，不影响容器溢出。

页面默认值

为提供更好的页面展示效果，Bootstrap v4 更新了 `<html>` and `<body>` 元素的一些属性，其中包括：

全局性地将每一个元素的 `box-sizing` 属性(包括 `*:before`、`*:after` 都设置为 `border-box` 以确保 DIV 元素自身定义的宽度不会因为 `border` 或 `padding` 而超过。

`<html>` 根元素没有声明 `font-size` 属性，但被假定为 `16px` 大小（这是目前 Chrome 等浏览器默认值），然后在此基础上采用 `font-size:1rem` 的比例应用于 `<body>` 上，使媒体查询能够轻松的实现缩放，最大程序保障用户偏好和易于访问。

`<body>` 元素被赋予一个全局性的 `font-family` 和 `line-height`，其下面的诸多表单元素也继承此属性，以防止字体大小错位冲突。

为了安全起见，`<body>` 的 `background-color` 的默认值赋为 `#fff`。

本地字体属性

Bootstrap 4 删除了默认的 Web 字体（Helvetica Neue, Helvetica 和 Arial），并替换为“本地 OS 字体引用机制”，以便在每个设备和操作系统上实现最佳文本呈现，参阅关于[本地字体引用](#)文章了解更多。

`$font-family-sans-serif:`

```
// Safari for OS X and iOS (San Francisco)
-apple-system,
// Chrome < 56 for OS X (San Francisco)
BlinkMacSystemFont,
// Windows
"Segoe UI",
// Android
"Roboto",
// Basic web fallback
"Helvetica Neue", Arial, sans-serif,
// Emoji fonts
"Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol" !default;
```

这样 `font-family` 适用于 `<body>`，并被全局并自动继承。切换全局 `font-family`，只要更新 `$font-family-base` 和重新编译 Bootstrap 即可。

标题和段落

所有标题和段落元素(如说<h1>以及<p>都被重置，系统移除它们的上外边距 **margin-top** 定义，标题添加外边距为 **margin-bottom: .5rem**，段落元素<p>添加了外边距 **margin-bottom:1rem** 以形成简洁行距。

Heading	Example
<h1></h1>	h1. Bootstrap heading
<h2></h2>	h2. Bootstrap heading
<h3></h3>	h3. Bootstrap heading
<h4></h4>	h4. Bootstrap heading
<h5></h5>	h5. Bootstrap heading
<h6></h6>	h6. Bootstrap heading

列表

移除所有的列表元素(、、 and <dl>) 的外边距 **margin-top**，并设置为 **margin-bottom: 1rem**，被嵌套的子列表没 **margin-bottom** 值。

- Lorem ipsum dolor sit amet
 - Consectetur adipiscing elit
 - Integer molestie lorem at massa
 - Facilisis in pretium nisl aliquet
 - Nulla volutpat aliquam velit
 - Phasellus iaculis neque
 - Purus sodales ultricies
 - Vestibulum laoreet porttitor sem
 - Ac tristique libero volutpat at
 - Faucibus porta lacus fringilla vel
 - Aenean sit amet erat nunc
 - Eget porttitor lorem
 1. Lorem ipsum dolor sit amet
 2. Consectetur adipiscing elit
 3. Integer molestie lorem at massa
 4. Facilisis in pretium nisl aliquet
 5. Nulla volutpat aliquam velit
 6. Faucibus porta lacus fringilla vel
 7. Aenean sit amet erat nunc
 8. Eget porttitor lorem

为了得到更简单的样式、清晰的等级以及更好的间距，描述列表 `code class="highlighter-rouge"><dd>`有一个更优级别的 **margin** 属性定义；其 **margin-left** 被重置为 0、并添加 **margin-bottom: .5rem** 值；另一方面<dt>的字体是**粗体**。

Description lists

A description list is perfect for defining terms.

Euismod

Vestibulum id ligula porta felis euismod semper eget lacinia odio sem.

Donec id elit non mi porta gravida at eget metus.

Malesuada porta

Etiam porta sem malesuada magna mollis euismod.

pre 预先格式化文本

`pre` 标签可定义预格式化的文本。被包围在 `pre`> 标签元素中的文本通常会保留空格和换行符。而文本也会呈现为等宽字体。

bootstrap 重置了 `pre` 元素，移除了它的 `margin-top` 属性并用 `rem` 作为 `margin-bottom` 的单位。

```
.example-element {  
  margin-bottom: 1rem;  
}
```

表格

微调了表格的样式，样式化了 `<caption>`，并确保 `text-align` 属性一致。与跟边框、内填充有关的细节，将在 [.table 表格](#) 一章门讲解。

This is an example table, and this is its caption to describe the contents.

Table heading	Table heading	Table heading	Table heading
Table cell	Table cell	Table cell	Table cell
Table cell	Table cell	Table cell	Table cell
Table cell	Table cell	Table cell	Table cell

Forms 表单

Bootstrap 重置了多种表单元素，得到简化的基本样式，使之简洁易用，显著变化表现在：

- `<fieldset>` 去除了边框、内填充、外边距属性，所以它们可以轻松地用作单一的输入框或者输入框组的放入容器中使用。
- `<legend>` 和 `fieldset` 字段集一样，也已被重新设计过，显示为不同种类的标题。
- `<label>` 加上了 `display: inline-block` 属性，从而可以被用户赋予 `margin` 属性进行布局调用。
- `<input>`、`<select>`、`<textarea>`s、`<button>` 基本本来都被规范化处理了，同时重置移除了它们的 `margin`，并且设置了 `inline-height: inherit` 属性。
- `<textarea>` 被修改为只能竖直方向上调整大小，因为水平方向上调整大小经常会“破坏”页面布局。

可以从下面方示例控件品味细节：

Example legend

Example input

Example
select

☐ Check this checkbox

☒ Option one is this and that ☐ Option two is something else that's also super long to demonstrate the wrapping of these fancy form controls. ☐ Option three is disabled



Example temporal

Example output 100

Button submit

Button submit

其它杂项

Address 地址控件

Bootstrap 更新了 `<address>` 元素初始属性，重置了浏览器默认的 `font-style`，由 `italic` 改为 `normal`、`line-height` 同样是继承来的，并添加了 `margin-bottom: 1rem`。`<address>` 是为最靠近根元素（或整个正文）提供联系信息。用结束行可保持格式。

Twitter, Inc.
1355 Market St, Suite 900
San Francisco, CA 94103
P: (123) 456-7890
Full Name
first.last@example.com

Blockquote 引用块效果

`blockquote` 引用块默认的 `margin` 是 `1em 40px`，而 Bootstrap 把它重置为 `0 0 1rem`，使其与其它元素更一致，下方示例：

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.

Someone famous in *Source Title*

addr 内联元素

`<abbr>` 内联元素接受基本的样式，使其在段落文本中突出。

Nulla attr vitae elit libero, a pharetra augue.

概要

`cursor` 摘要上的默认值是 `text`，所以我们重置，以 `pointer` 通过单击它来传达该元素可以进行交互。这里有更详细的细节。

HTML5 的 `[hidden]` 属性

HTML5 标准中增加了一个 [全新的 `\[hidden\]` 标签](#)，它的默认值被赋予为 `display: none`，借鉴 [PureCSS](#) 的思路，我们将它重定义为 `[hidden] { display: none !important; }`，以防止它的 `display` 值被意外覆盖，即使 IE10 不支持 `[hidden]` 属性，但通过引用 Bootstrap 的引入明确解决了此问题。

```
<input type="text" hidden>
```

jQuery 冲突

`[hidden]` 与 jQuery 的两个方法不兼容，分别是：`$(...).hide()`、`$(...).show()`，目前还没有特别好的 `[hidden]` 其他管理 `display` 元素的技术。

仅仅切换元素的可见性，这意味着其 `display` 不被修改和元素还会影响文档的流程，使用 `.invisible` 类来代替。

点击触摸的延迟优化

传统上，触摸屏设备上的浏览器在“点击”结束（手指/手写笔从屏幕提起的那一刻）和 `click` 事件 被触发之间延迟大约 300ms。这种延迟对于这些浏览器正确处理“双击放大”手势是必要的，而不会在第一个“点击”之后过早触发动作或链接，但它可能会使您的网站感到轻微迟钝和无响应。

大多数移动浏览器会自动优化远程使用该 `width=device-width` 属性的站点的 300ms 延迟，作为其响应元标记的一部分（以及禁用缩放的站点，例如 `user-scalable=no`，尽管这种做法对于可访问性和可用性原因非常不鼓励）。这里最大的例外是 Windows Phone 8.1 上的 IE11 和 iOS 9.3 之前的 iOS Safari（以及任何其他基于 iOS WebView 的浏览器）。

在触摸式笔记本电脑/桌面设备上，IE11 和 Microsoft Edge 目前是唯一具有“双击缩放”功能的浏览器。由于响应式元标记被所有桌面浏览器忽略，所以使用 `width=device-width` 将对这里的 300ms 延迟没有影响。

为了在 IE11 和桌面上的 Microsoft Edge 以及 Windows Phone 8.1 上的 IE11 中解决此问题，Bootstrap 显式地在所有交互式元素（如按钮和链接）上使用 `touch-action:manipulation` [CSS 私有属性](#)。该属性基本上禁用了这些元素的双击功能，从而消除了 300ms 的延迟。

在旧的 iOS 版本（9.3 之前）的情况下，建议的方法是使用其他脚本，如 [FastClick](#) 来明确解决延迟。

排版

Bootstrap 排版的文档和示例，包括全局设置、标题、正文列表等。

全局设置

Bootstrap 定义了基本的全局显示、排版、以及链接样式，同时提供了一个 [通用文本样式示例](#)。

- 使用 [本地字体堆栈](#)，从而为每一个操作系统或设备上的 `font-family` 渲染都得到最佳的体现（而不是强制多设备共享一套字体呈现标准）。
- 对于更具包容性和可访问的类型规模，我们假设浏览器默认根 `font-size`（通常为 16 像素），而访客根据自身需要定义浏览器字体大小呈现，并不会影响网页表现。
- Use the `$font-family-base`, `$font-size-base`, and `$line-height-base` attributes as our typographic base applied to the `<body>`.
- 使用 `code class="highlighter-rouge">$link-color` 参数定义全局的 `a` 链接颜色和 `:hover` 下划线颜色呈现。
- 使用 `$body-bg` 参数定义 `<body>` 的 `background-color` 属性，默认认为白色 (`#fff`)。

这些定义可以在 `_reboot.scss` 找到，并根据需要自定义全局变量 `_variables.scss`（确保 `$font-size-base` 使用 `rem` 单位。

标题

兼容所有 HTML 标题集，涵括从 `<h1>` 到 `<h6>` 的六种标题展现。

标签	效果
<code><h1></h1></code>	h1. Bootstrap heading
<code><h2></h2></code>	h2. Bootstrap heading
<code><h3></h3></code>	h3. Bootstrap heading
<code><h4></h4></code>	h4. Bootstrap heading
<code><h5></h5></code>	h5. Bootstrap heading
<code><h6></h6></code>	h6. Bootstrap heading

Copy

```
<h1>h1. Bootstrap heading</h1>
<h2>h2. Bootstrap heading</h2>
<h3>h3. Bootstrap heading</h3>
<h4>h4. Bootstrap heading</h4>
<h5>h5. Bootstrap heading</h5>
<h6>h6. Bootstrap heading</h6>
```

CSS 选择器也支持以 `.h1` -- `.h6` 方式引用，这样可以使字体样式呈现出标题效果，不同是引用 `.h1` -- `.h6` 的文本段不会视作 HTML 的标题元素（往往 SEO、读屏器和机器识别时对此很敏感-译者注），效果如下：

h1. Bootstrap heading

h2. Bootstrap heading

h3. Bootstrap heading

h4. Bootstrap heading

h5. Bootstrap heading

h6. Bootstrap heading

```
<p class="h1">h1. Bootstrap heading</p>
<p class="h2">h2. Bootstrap heading</p>
<p class="h3">h3. Bootstrap heading</p>
<p class="h4">h4. Bootstrap heading</p>
```



```
<p class="h5">h5. Bootstrap heading</p>
<p class="h6">h6. Bootstrap heading</p>
```

自定义标题备注

使用附带的实用类从 Bootstrap 重新创建小的辅助标题文本，如下所示：

Fancy display heading With faded secondary text

Copy

```
<h3>
  Fancy display heading
  <small class="text-muted">With faded secondary text</small>
</h3>
```

显式标题

bootstrap 可以传统的标题元素设计得更漂亮，以迎合你的网页内容。如果你想要一个标题醒目，考虑使用显示标题——一更大型、鲜明的标题样式，则可以用下面方法：

```
<h1 class="display-1">Display 1</h1>
<h1 class="display-2">Display 2</h1>
<h1 class="display-3">Display 3</h1>
<h1 class="display-4">Display 4</h1>
```

Lead 中心内容

通过应用 `.lead` 样式，可以定义一个中心段落，用于提示这是中心内容或重要内容。

Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis, est non commodo luctus.

Copy

```
<p class="lead">
  Vivamus sagittis lacus vel augue laoreet rutrum faucibus dolor auctor. Duis mollis,
  est non commodo luctus.
</p>
```

文本内联元素

HTML5 文本元素的常用内联表现方法也适用于 Bootstrap4。

You can use the mark tag to highlight text.

This line of text is meant to be treated as deleted text.

~~This line of text is meant to be treated as no longer accurate.~~

This line of text is meant to be treated as an addition to the document.

This line of text will render as underlined

This line of text is meant to be treated as fine print.

This line rendered as bold text.

This line rendered as italicized text.

Copy

```
<p>You can use the mark tag to <mark>highlight</mark> text.</p>
<p><del>This line of text is meant to be treated as deleted text.</del></p>
<p><s>This line of text is meant to be treated as no longer accurate.</s></p>
<p><ins>This line of text is meant to be treated as an addition to the
document.</ins></p>
<p><u>This line of text will render as underlined</u></p>
<p><small>This line of text is meant to be treated as fine print.</small></p>
<p><strong>This line rendered as bold text.</strong></p>
<p><em>This line rendered as italicized text.</em></p>
```

`.mark`、`.small` 类也可以应用相同的 HTML 标签 `<mark>`、`<small>`，这样还可以避免标签带来的任何不必要的语义含义。虽然上面没有明确显示，但我们可以随意使用 `` 和 `<i>` 等 HTML5 标签，其中 `` 旨在突出显示单词或短语、而不会增加重要性，`<i>` 主要用于语音、技术术语等。

文本实用程序

使用 Bootstrap 提供的 [文本实用程序](#) 可更改文本对齐，变换，样式，权重和颜色。

abbr 缩略语

Stylized implementation of HTML's `<abbr>` element for abbreviations and acronyms to show the expanded version on hover. Abbreviations have a default underline and gain a help cursor to provide additional context on hover and to users of assistive technologies.

Add `.initialism` to an abbreviation for a slightly smaller font-size.

attr

HTML

Copy

```
<p><abbr title="attribute">attr</abbr></p>
<p><abbr title="HyperText Markup Language" class="initialism">HTML</abbr></p>
```

blockquote 来源备注与引用

引用文档中另一个来源的内容块，，请在一段 HTML 外面包裹 `<blockquote class="blockquote">`，作为引用。为了显示直接引用，我们推荐使用 `p` 作为子级包裹容器，这在 HTML 规范中也有多个变通方法，下面逐一讲解。

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.

Copy

```
<blockquote class="blockquote">
  <p class="mb-0">Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.</p>
</blockquote>
```

底部备注来源

`<footer class="blockquote-footer">` 用于标识来源，一般用于页脚（所以有`*-footer`），然后配合 `<cite>`使用。

爱上一个地方，就应该背上包去旅行，走得更远。

出自商务印书馆的 《新华字典》

Copy

```
<blockquote class="blockquote">
  <p class="mb-0">爱上一个地方，就应该背上包去旅游，走得更远.</p>
  <footer class="blockquote-footer">出自商务印书馆的 <cite title="Source Title">《新华字典》</cite></footer>
</blockquote>
```

对齐处理

如果需要居中对齐或右对齐，使用`.text-center`、`.text-right` 方法配合即可，如下两示例：

爱上一个地方，就应该背上包去旅行，走得更远。

```
<blockquote class="blockquote text-center">
  <p class="mb-0">爱上一个地方，就应该背上包去旅行，走得更远.</p>
  <footer class="blockquote-footer">出自商务印书馆的 <cite title="Source Title">《新华字典》</cite></footer>
</blockquote>
```

爱上一个地方，就应该背上包去旅行，走得更远。

```
<blockquote class="blockquote text-right">
  <p class="mb-0">爱上一个地方，就应该背上包去旅行，走得更远.</p>
  <footer class="blockquote-footer">出自商务印书馆的 <cite title="Source Title">《新华字典》</cite></footer>
</blockquote>
```

列表

列表样式初始化

在 ul(或 ol) 上使用 `.list-unstyled` 可以删除列表项目上默认的 `list-style` 以及左外边距（只针对直接子元素），这只生效于在直接子列表项目上，不影响你嵌套的子列表。

- Lorem ipsum dolor sit amet
- Consectetur adipiscing elit
- Integer molestie lorem at massa
- Facilisis in pretium nisl aliquet
- Nulla volutpat aliquam velit
 - Phasellus iaculis neque
 - Purus sodales ultricies
 - Vestibulum laoreet porttitor sem
 - Ac tristique libero volutpat at
- Faucibus porta lacus fringilla vel
- Aenean sit amet erat nunc
- Eget porttitor lorem

Copy

```
<ul class="list-unstyled">
  <li>Lorem ipsum dolor sit amet</li>
  <li>Consectetur adipiscing elit</li>
  <li>Integer molestie lorem at massa</li>
  <li>Facilisis in pretium nisl aliquet</li>
  <li>Nulla volutpat aliquam velit
    <ul>
      <li>Phasellus iaculis neque</li>
      <li>Purus sodales ultricies</li>
      <li>Vestibulum laoreet porttitor sem</li>
      <li>Ac tristique libero volutpat at</li>
    </ul>
  </li>
  <li>Faucibus porta lacus fringilla vel</li>
  <li>Aenean sit amet erat nunc</li>
  <li>Eget porttitor lorem</li>
</ul>
```

分行或单行多列并排

使用 `.list-inline`、`.list-inline-item` 结合，可以实现列表逐行显示（默认不引用且无父元素影响也是逐行显示）、或单行并多列并排（遵循从左对右的原则、并清除 `margin` 方法）。

- 列表之一
- 列表之二
- 列表之三

```
<ul class="list-inline">
  <li class="list-inline">列表之一</li>
  <li class="list-inline">列表之二</li>
  <li class="list-inline">列表之三</li>
</ul>
```

```
<ul class="list-inline">
  <li class="list-inline-item">列表之一</li>
  <li class="list-inline-item">列表之二</li>
  <li class="list-inline-item">列表之三</li>
</ul>
```

dl 表格式水平描述

使用 Bootstrap 栅格系统的预定义类（或者说语义化 mixins），可以水平对齐条目和描述。对于较长的条目，你可以视情况添加一个 `.text-truncate` 类，从而用省略号截断文本。

描述列表

一个关于描述列表的两端对齐

Euismod

Vestibulum id ligula porta felis euismod semper eget lacinia odio sem nec elit.

Donec id elit non mi porta gravida at eget metus.

Malesuada porta

Etiam porta sem malesuada magna mollis euismod.

Truncated term is truncated

Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.

Nesting

Nested definition list

Aenean posuere, tortor sed cursus feugiat, nunc augue blandit nunc.

```
<dl class="row">
  <dt class="col-sm-3">描述列表</dt>
  <dd class="col-sm-9">一个关于描述列表的两端对齐</dd>

  <dt class="col-sm-3">Euismod</dt>
  <dd class="col-sm-9">
    <p>Vestibulum id ligula porta felis euismod semper eget lacinia odio sem nec elit.</p>
    <p>Donec id elit non mi porta gravida at eget metus.</p>
  </dd>

  <dt class="col-sm-3">Malesuada porta</dt>
  <dd class="col-sm-9">Etiam porta sem malesuada magna mollis euismod.</dd>

  <dt class="col-sm-3 text-truncate">Truncated term is truncated</dt>
  <dd class="col-sm-9">Fusce dapibus, tellus ac cursus commodo, tortor mauris condimentum nibh, ut fermentum massa justo sit amet risus.</dd>

  <dt class="col-sm-3">Nesting</dt>
  <dd class="col-sm-9">
    <dl class="row">
      <dt class="col-sm-4">Nested definition list</dt>
```

```
<dd class="col-sm-8">Aenean posuere, tortor sed cursus feugiat, nunc augue  
blandit nunc.</dd>  
</dl>  
</dd>  
</dl>
```

响应式排版

响应性排版是指通过简单地调整 font-size 一系列媒体查询中的根元素来缩放文本和组件。Bootstrap 不会为您做这个，但是如果需要，它很容易添加。这是实践中的一个例子。选择 font-size 您想要的任何 s 和媒体查询。

```
html {  
  font-size: 1rem;  
}  
  
@include media-breakpoint-up(sm) {  
  html {  
    font-size: 1.2rem;  
  }  
}  
  
@include media-breakpoint-up(md) {  
  html {  
    font-size: 1.4rem;  
  }  
}  
  
@include media-breakpoint-up(lg) {  
  html {  
    font-size: 1.6rem;  
  }  
}
```

代码

基于 Bootstrap 显示行内嵌入的内联代码和多行代码段的文档和示例。

内联代码

用 `<code>` 包裹内联代码片断，勿忘转义 HTML 尖括号。

示例： `<section>` 代码嵌入到文本段中。

Copy

示例： `<code><section></code>` 代码嵌入到文本段中。

代码块

使用 `<pre>` 标签可以包裹代码块，同样 HTML 的尖括号需要进行转义，你还可以使用 `.pre-scrollable` CSS 样式，实现垂直滚动的效果，它默认提供 350px 高度限制、Y 轴垂直滚动效果。

`<p>Sample text here...</p>`

`<p>And another line of sample text here...</p>`

Copy

```
<pre><code>&lt;p&gt;Sample text here...&lt;/p&gt;
&lt;p&gt;And another line of sample text here...&lt;/p&gt;
</code></pre>
```

`<p>Sample text here...</p>`

`<p>引用.pre-scrollable 实现限高垂直滚动`

`</p>`

Copy

```
<pre><code>&lt;p&gt;引用.pre-scrollable 实现限高垂直滚动
<br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>
<br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>&lt;/p&gt;
&lt;p&gt;And another line of sample text here...&lt;/p&gt;
</code></pre>
```

Var 变量

推荐使用 `<var>` 标签包裹标示变量。

$y = mx + b$

Copy

```
<var>y</var> = <var>m</var><var>x</var> + <var>b</var>
```

用户输入（键盘动作提示）

使用 `<kbd>` 标签，标明这是一个键盘输入操作。

To switch directories, type followed by the name of the directory.

To edit settings, press

Copy

To switch directories, type `<kbd>cd</kbd>` followed by the name of the directory.`
`

To edit settings, press `<kbd><kbd>ctrl</kbd> + <kbd>,</kbd></kbd>`

示例标注

`<samp>` 标签代表这是一个示例。

这是一个代码示例。

Copy

```
<samp>这是一个代码示例.</samp>
```

图片

Bootstrap 图片处理的示例和文档，由于我们为图片添加了轻量级的无干扰样式和响应式行为，因此在 Bootstrap 设计中引用图片可以更加方便且不会轻易撑破其它元素。

响应式图片

在 Bootstrap 中，给图片添加 `.img-fluid` 样式，或定义 `max-width: 100%`、`height: auto;` 样式，即可赋予响应式特性，图片大小会随着父元素大小同步缩放。

```

```

IE 10 问题以及 SVG 图形的特殊处理

在 IE 10 浏览器中，带 `.img-responsive` 类的 SVG 图片尺寸可能会不均称，这是一个浏览器级的缺陷，你可以在相应图片元素上添加 `width: 100% \9` 来解决它（`width: 100% \9` 方法不能普遍引用，否则会造成其它图片格式的混乱，所以 Bootstrap 没有全局自动引用之）。

缩略图处理

您可以使用 `.img-thumbnail` 属性来使图片自动被加上一个带圆角且 1px 边界的外框缩略图样式（你也可以使用系统提供的边框间距方法，如 `.p-1` 再加上边框颜色定义达成），效果如下：

```

```

图像对齐处理

对于 `.block` 属性的块状图片，我们也可以使用 [浮动定义规范](#) 或 [文字对齐规范](#)，来实现对图像的对齐、浮动控制，带 `.block` 块属性的图片，可以自动获得 `.mx-auto` 的位置对齐属性。

```





<div class="text-center">
  
</div>
```


Html 5 标准之 Picture 元素

HTML5 标准提供了一个全新的<picture> 元素，它可以为 指定多个<source> 定义，请确保在 标签里使用使用 .img-* CSS 样式进行定义绑定，而不是仅仅认为引用了 就达成了，如下第三行代码所示：

```
<picture>
  <source srcset="..." type="image/svg+xml">
  
</picture>
```

<picture> 元素可实现图片在不同屏幕下的针对性响应式，其使用逻辑如下(译者补充)：

Copy

```
<picture>
  <source src="大规格图片.jpg" media="(min-width: 800px)" >
  <source src="中规格图片.jpg" media="(min-width: 600px)">
  <source src="小规格图片.jpg">
  
</picture>
```

表格

使用 Bootstrap 使用表格的文档和示例（表格样式及响应式优先于 Bootstrap 全局的其它 JavaScript 事件）。

示例

在第三方部件例如日历和日期选择器中广泛使用表格，我们设计了视情况需要加入的表格类。只需要向某个<table>添加一个基类 .table，然后通过自定义样式或系统提供的 class 来起作用。

使用最基本的表格标记，下面是 Bootstrap 中 .table 表格的样式（基本样式）， **Bootstrap 4 继承了所有的表格样式**，这意味着任何嵌套表格都将以与父类型相同的方式进行样式化。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

```
<table class="table">
  <thead>
```

```

<tr>
  <th scope="col">#</th>
  <th scope="col">First Name</th>
  <th scope="col">Last Name</th>
  <th scope="col">Username</th>
</tr>
</thead>
<tbody>
  <tr>
    <th scope="row">1</th>
    <td>Mark</td>
    <td>Otto</td>
    <td>@mdo</td>
  </tr>
  <tr>
    <th scope="row">2</th>
    <td>Jacob</td>
    <td>Thornton</td>
    <td>@fat</td>
  </tr>
  <tr>
    <th scope="row">3</th>
    <td>Larry</td>
    <td>the Bird</td>
    <td>@twitter</td>
  </tr>
</tbody>
</table>

```

你可使用 `.table-dark` class 选择器来产生颜色反转对比效果，即深色背景和浅色文本。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Copy

```

<table class="table table-dark">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>

```

```

    <th scope="col">Username</th>
  </tr>
</thead>
<tbody>
  <tr>
    <th scope="row">1</th>
    <td>Mark</td>
    <td>Otto</td>
    <td>@mdo</td>
  </tr>
  <tr>
    <th scope="row">2</th>
    <td>Jacob</td>
    <td>Thornton</td>
    <td>@fat</td>
  </tr>
  <tr>
    <th scope="row">3</th>
    <td>Larry</td>
    <td>the Bird</td>
    <td>@twitter</td>
  </tr>
</tbody>
</table>
```

Head 表头处理

与预设的反转样式相似，使用 `.thead-light` 或 `.thead-dark` 中的一个样式，就能使 `<thead>` 区显示出浅黑或深灰。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Copy

```
<table class="table">
```

```

<thead class="thead-dark">
  <tr>
    <th scope="col">#</th>
    <th scope="col">First Name</th>
    <th scope="col">Last Name</th>
    <th scope="col">Username</th>
  </tr>
</thead>
<tbody>
  <tr>
    <th scope="row">1</th>
    <td>Mark</td>
    <td>Otto</td>
    <td>@mdo</td>
  </tr>
  <tr>
    <th scope="row">2</th>
    <td>Jacob</td>
    <td>Thornton</td>
    <td>@fat</td>
  </tr>
  <tr>
    <th scope="row">3</th>
    <td>Larry</td>
    <td>the Bird</td>
    <td>@twitter</td>
  </tr>
</tbody>
</table>

```

```

<table class="table">
  <thead class="thead-light">
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>

```

```

        <td>Mark</td>
        <td>Otto</td>
        <td>@mdo</td>
    </tr>
    <tr>
        <th scope="row">2</th>
        <td>Jacob</td>
        <td>Thornton</td>
        <td>@fat</td>
    </tr>
    <tr>
        <th scope="row">3</th>
        <td>Larry</td>
        <td>the Bird</td>
        <td>@twitter</td>
    </tr>
</tbody>
</table>

```

条纹状表格

使用 `.table-striped` 样式定义 `<tbody>`，可以产生逐行颜色强烈对比的表格样式（以及增加反转）。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Copy

```

<table class="table table-striped">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>

```

```

        <td>Otto</td>
        <td>@mdo</td>
    </tr>
    <tr>
        <th scope="row">2</th>
        <td>Jacob</td>
        <td>Thornton</td>
        <td>@fat</td>
    </tr>
    <tr>
        <th scope="row">3</th>
        <td>Larry</td>
        <td>the Bird</td>
        <td>@twitter</td>
    </tr>
</tbody>
</table>

```

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

Copy

```

<table class="table table-striped table-dark">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Jacob</td>
      <td>Thornton</td>

```

```

        <td>@fat</td>
    </tr>
    <tr>
        <th scope="row">3</th>
        <td>Larry</td>
        <td>the Bird</td>
        <td>@twitter</td>
    </tr>
</tbody>
</table>

```

Bordered table

添加 `.table-bordered` 类可以产生表格边框与间隙系统。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Mark	Otto	@TwBootstrap
3	Jacob	Thornton	@fat
4	Larry the Bird		@twitter

Copy

```

<table class="table table-bordered">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Mark</td>

```

```

        <td>Otto</td>
        <td>@TwBootstrap</td>
    </tr>
    <tr>
        <th scope="row">3</th>
        <td>Jacob</td>
        <td>Thornton</td>
        <td>@fat</td>
    </tr>
    <tr>
        <th scope="row">4</th>
        <td colspan="2">Larry the Bird</td>
        <td>@twitter</td>
    </tr>
</tbody>
</table>

```

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Mark	Otto	@TwBootstrap
3	Jacob	Thornton	@fat
4	Larry the Bird		@twitter

Copy

```

<table class="table table-bordered table-dark">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Mark</td>
      <td>Otto</td>

```



```

        <td>@TwBootstrap</td>
    </tr>
    <tr>
        <th scope="row">3</th>
        <td>Jacob</td>
        <td>Thornton</td>
        <td>@fat</td>
    </tr>
    <tr>
        <th scope="row">4</th>
        <td colspan="2">Larry the Bird</td>
        <td>@twitter</td>
    </tr>
</tbody>
</table>

```

行悬停效果

将 `.table-hover` 定义到 `<tbody>` 上，可以产生行悬停效果（鼠标移到行上会出现状态提示）。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

Copy

```

<table class="table table-hover">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>

```

```

<tr>
  <th scope="row">2</th>
  <td>Jacob</td>
  <td>Thornton</td>
  <td>@fat</td>
</tr>
<tr>
  <th scope="row">3</th>
  <td colspan="2">Larry the Bird</td>
  <td>@twitter</td>
</tr>
</tbody>
</table>

```

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

Copy

```

<table class="table table-hover table-dark">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Jacob</td>
      <td>Thornton</td>
      <td>@fat</td>
    </tr>
    <tr>
      <th scope="row">3</th>

```

```
        <td colspan="2">Larry the Bird</td>
        <td>@twitter</td>
    </tr>
</tbody>
</table>
```

紧缩表格

添加 `.table-sm` 可以将表格的 `padding` 值缩减一半，使表格更加紧凑。

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

Copy

```
<table class="table table-sm">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Jacob</td>
      <td>Thornton</td>
      <td>@fat</td>
    </tr>
    <tr>
      <th scope="row">3</th>
      <td colspan="2">Larry the Bird</td>
      <td>@twitter</td>
```

```
</tr>
</tbody>
</table>
```

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry the Bird		@twitter

Copy

```
<table class="table table-sm table-dark">
  <thead>
    <tr>
      <th scope="col">#</th>
      <th scope="col">First Name</th>
      <th scope="col">Last Name</th>
      <th scope="col">Username</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th scope="row">1</th>
      <td>Mark</td>
      <td>Otto</td>
      <td>@mdo</td>
    </tr>
    <tr>
      <th scope="row">2</th>
      <td>Jacob</td>
      <td>Thornton</td>
      <td>@fat</td>
    </tr>
    <tr>
      <th scope="row">3</th>
      <td colspan="2">Larry the Bird</td>
      <td>@twitter</td>
    </tr>
  </tbody>
</table>
```

语义状态化

使用语义状态样式对表格逐行或单个单元格进行着色表达。

Type	Column heading	Column heading	Column heading
Active	Column content	Column content	Column content
Default	Column content	Column content	Column content
Primary	Column content	Column content	Column content
Secondary	Column content	Column content	Column content
Success	Column content	Column content	Column content
Danger	Column content	Column content	Column content
Warning	Column content	Column content	Column content
Info	Column content	Column content	Column content
Light	Column content	Column content	Column content
Dark	Column content	Column content	Column content

Copy

```
<!-- On rows -->
<tr class="table-active">...</tr>

<tr class="table-primary">...</tr>
<tr class="table-secondary">...</tr>
<tr class="table-success">...</tr>
<tr class="table-danger">...</tr>
<tr class="table-warning">...</tr>
<tr class="table-info">...</tr>
<tr class="table-light">...</tr>
<tr class="table-dark">...</tr>

<!-- On cells (`td` or `th`) -->
<tr>
  <td class="table-active">...</td>

  <td class="table-primary">...</td>
  <td class="table-secondary">...</td>
  <td class="table-success">...</td>
  <td class="table-danger">...</td>
  <td class="table-warning">...</td>
  <td class="table-info">...</td>
  <td class="table-light">...</td>
  <td class="table-dark">...</td>
</tr>
```

深色表格上没有固定的背景，你可以使用 [文字或背景通用样式](#) 获得类似的样式：

#	Column heading	Column heading	Column heading
1	Column content	Column content	Column content
2	Column content	Column content	Column content
3	Column content	Column content	Column content
4	Column content	Column content	Column content
5	Column content	Column content	Column content

#	Column heading	Column heading	Column heading
6	Column content	Column content	Column content
7	Column content	Column content	Column content
8	Column content	Column content	Column content
9	Column content	Column content	Column content

Copy

```
<!-- On rows -->
<tr class="bg-primary">...</tr>
<tr class="bg-success">...</tr>
<tr class="bg-warning">...</tr>
<tr class="bg-danger">...</tr>
<tr class="bg-info">...</tr>

<!-- On cells (`td` or `th`) -->
<tr>

  <td class="bg-primary">...</td>
  <td class="bg-success">...</td>
  <td class="bg-warning">...</td>
  <td class="bg-danger">...</td>
  <td class="bg-info">...</td>
</tr>
```

向使用辅助技术的用户传达用意

使用颜色添加意义只提供一个视觉指示，这不会传达给辅助技术的用户，如屏幕阅读器。确保由颜色表示的信息从内容本身（例如可见文本）中显而易见，或者通过替代方法包括，例如隐藏在.sr-only 该类中的附加文本。

Captions 表格辅助标题

<caption> 标签如同一个表格的标题，它默认是隐藏的，可以协助屏幕阅读器用户找到表格、了解表格内容，且决定是否需要阅读它。

List of users

#	First Name	Last Name	Username
1	Mark	Otto	@mdo
2	Jacob	Thornton	@fat
3	Larry	the Bird	@twitter

```
<table class="table">
  <caption>List of users</caption>
  <thead>
    <tr>
      <th scope="col">#</th>
```


#	Heading	Heading	Heading	Heading	Heading	Heading	Heading
3	Cell	Cell	Cell	Cell	Cell	Cell	Cell

```
<div class="table-responsive">
  <table class="table">
    ...
  </table>
</div>
```

多屏幕断点设定

使用 `.table-responsive{-sm|-md|-lg|-xl}` 可以根据需要对不同屏幕尺寸断点表格进行响应式设计，在中断点之上该表格会正常显示，而不是水平滚动（没有水平条）。

#	Heading	Heading	Heading	Heading	
1	Cell	Cell	Cell	Cell	((: :
2	Cell	Cell	Cell	Cell	((: :
3	Cell	Cell	Cell	Cell	((: :
#	Heading	Heading	Heading	Heading	
1	Cell	Cell	Cell	Cell	((: :
2	Cell	Cell	Cell	Cell	((: :
3	Cell	Cell	Cell	Cell	(

					1
					6
					8
					0
					:
					1
#	Heading	Heading	Heading	Heading	}
					6
					:
					:

					1
					6
					8
					0
					:
					1
#	Heading	Heading	Heading	Heading	}
1	Cell	Cell	Cell	Cell	(
					6
					:
					:
2	Cell	Cell	Cell	Cell	(
					6
					:
					:
3	Cell	Cell	Cell	Cell	(
					6
					:
					:

					1
					6
					8
					0
					:
					1
#	Heading	Heading	Heading	Heading	}
1	Cell	Cell	Cell	Cell	(
					6
					:
					:
2	Cell	Cell	Cell	Cell	(
					6
					:
					:
3	Cell	Cell	Cell	Cell	(
					6
					:
					:

Copy

```
<figure class="figure">
  
  <figcaption class="figure-caption">A caption for the above image.</figcaption>
</figure>
```

文字对齐控制

结合我们的[文本实用工具](#)可以轻松对齐图文框的说明文字（比如右对齐、左对齐，只要引用 `.text-*` 方法即可-译者注）。

A caption for the above image.

Copy

```
<figure class="figure">
  
  <figcaption class="figure-caption text-right">A caption for the above image.</figcaption>
</figure>
```

警告提示框(Alert)

警告框组件通过提供一些灵活的预定义消息，为常见的用户动作提供常见的上下反馈消息和提示。

示例

警报是一组颜色控件（共八个颜色样式），可用于任何长度的文本，以及可选的关闭按钮，系统提供 8 个可用的正确的样式（如，`.alert-success`），如你还可以使用 [jQuery 插件方法](#)来解除内联元素。

This is a primary alert—check it out!

This is a secondary alert—check it out!

This is a success alert—check it out!

This is a danger alert—check it out!

This is a warning alert—check it out!

This is a info alert—check it out!

This is a light alert—check it out!

This is a dark alert—check it out!

```
<div class="alert alert-primary" role="alert">
  This is a primary alert—check it out!
</div>
<div class="alert alert-secondary" role="alert">
  This is a secondary alert—check it out!
```

```

</div>
<div class="alert alert-success" role="alert">
  This is a success alert—check it out!
</div>
<div class="alert alert-danger" role="alert">
  This is a danger alert—check it out!
</div>
<div class="alert alert-warning" role="alert">
  This is a warning alert—check it out!
</div>
<div class="alert alert-info" role="alert">
  This is a info alert—check it out!
</div>
<div class="alert alert-light" role="alert">
  This is a light alert—check it out!
</div>
<div class="alert alert-dark" role="alert">
  This is a dark alert—check it out!
</div>

```

网页中传达辅助技术及其背后的意义

使用颜色添加意义只提供一个视觉指示，这不会传达给需要辅助技术（如盲人、听力障碍者）的用户，也就决定了诸如屏幕阅读器并不能读出颜色本身的意义。一般推荐确保由颜色表示的信息从内容本身（例如可见文本）中显而易见，或者通过替代方法，例如隐藏在 `.sr-only` 该类中的附加文本来创造更多的辅助传达技术。

`.sr-only`，全称 *screen reader only*，意为：(仅供)屏幕阅读器，这个 class 主要用于增强 accessibility(可访问性)，有时候 UI 上会出现一些仅供视觉识别的元素，比如说“菜单按钮”，只有视力正常的人才能清楚辨识这些元素的作用。而残障人士，比如弱势或盲人是不可能知道这些视觉识别元素是什么的。他们上网使用的是屏幕阅读器，也就是 screen reader (sr)，屏幕阅读器需要找到能辨识的文本说明然后“读”出来给用户听。问题是图形元素怎么可能“读出来”呢？因此我们还要写上这些元素的文本说明，但是又不需要展示给普通用户看到，于是加上 `sr-only` 的意义就在于能保证屏幕阅读器正确读取且不会影响 UI 的视觉呈现（译者补充）。

链接颜色

使用 `.alert-link` 类可以为带颜色的警告文本框中的链接加上合适的颜色（Bootstrap 已经内置了相应的颜色解决方案，会自动对应有一个优化后的链接颜色方案-译者注）。

```

This is a primary alert with an example link. Give it a click if you like.
This is a secondary alert with an example link. Give it a click if you like.
This is a success alert with an example link. Give it a click if you like.
This is a danger alert with an example link. Give it a click if you like.
This is a warning alert with an example link. Give it a click if you like.
This is a info alert with an example link. Give it a click if you like.
This is a light alert with an example link. Give it a click if you like.
This is a dark alert with an example link. Give it a click if you like.

```

Copy

```

<div class="alert alert-primary" role="alert">
  This is a primary alert with <a href="#" class="alert-link">an example link</a>.
  Give it a click if you like.

```

```
</div>
<div class="alert alert-secondary" role="alert">
  This is a secondary alert with <a href="#" class="alert-link">an example link</a>.
  Give it a click if you like.
</div>
<div class="alert alert-success" role="alert">
  This is a success alert with <a href="#" class="alert-link">an example link</a>.
  Give it a click if you like.
</div>
<div class="alert alert-danger" role="alert">
  This is a danger alert with <a href="#" class="alert-link">an example link</a>.
  Give it a click if you like.
</div>
<div class="alert alert-warning" role="alert">
  This is a warning alert with <a href="#" class="alert-link">an example link</a>.
  Give it a click if you like.
</div>
<div class="alert alert-info" role="alert">
  This is a info alert with <a href="#" class="alert-link">an example link</a>. Give
  it a click if you like.
</div>
<div class="alert alert-light" role="alert">
  This is a light alert with <a href="#" class="alert-link">an example link</a>.
  Give it a click if you like.
</div>
<div class="alert alert-dark" role="alert">
  This is a dark alert with <a href="#" class="alert-link">an example link</a>. Give
  it a click if you like.
</div>
```

额外附加内容

警报还可以包含其他 HTML 元素，如标、段落和分隔符。

Well done!

Aww yeah, you successfully read this important alert message. This example text is going to run a bit longer so that you can see how spacing within an alert works with this kind of content.

Whenever you need to, be sure to use margin utilities to keep things nice and tidy.

Copy

```
<div class="alert alert-success" role="alert">
  <h4 class="alert-heading">Well done!</h4>
```

```
<p>Aww yeah, you successfully read this important alert message. This example text is going to run a bit longer so that you can see how spacing within an alert works with this kind of content.</p>
<hr>
<p class="mb-0">Whenever you need to, be sure to use margin utilities to keep things nice and tidy.</p>
</div>
```

关闭警告（小贴士效果）

使用 `.alert` 结合 JavaScript，可以实现警报效果，贴在页面上，并可以自由关闭，其要点如下：

- 确保你正确加载了 `.alert` 警报组件，或者是编译后的 Bootstrap JavaScript 代码组。
- 如果你要自行编译 JavaScript 组件，请将[必须的](#) `util.js` 包括进去。
- 可以在右上角定义一个 `.close` 关闭按钮效果，则需要在容器中引用 `.alert-dismissible` 类。
- 警告按钮上要增加 `data-dismiss="alert"` 触发 JavaScript 动作，同时使用 `<button>` 元素，以确保在所有设备上都能获得正确的行为响应。
- 要在关闭警报时生成警报提示，请确保添加 `.fade` 和 `.show` 样式。

实际效果展示：

Holy guacamole! You should check in on some of those fields below.x

Copy

```
<div class="alert alert-warning alert-dismissible fade show" role="alert">
  <strong>Holy guacamole!</strong> You should check in on some of those fields below.
  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
    <span aria-hidden="true">&times;</span>
  </button>
</div>
```

JavaScript 行为

触发

使用 JavaScript 插件解除警报：

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

或者使用 **警报控件** 中的 `data` 属性，如下所示：

Copy

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

请注意，关闭警报会将其从 DOM 中移除。

方法

方法	描述
<code>\$().alert()</code>	Makes an alert listen for click events on descendant elements which have the <code>data-dismiss="alert"</code> attribute. (Not necessary when using the data-api's auto-initialization.)
<code>\$().alert('close')</code>	Closes an alert by removing it from the DOM. If the <code>.fade</code> and <code>.show</code> classes are present on the element, the alert will fade out before it is removed.
<code>\$().alert('dispose')</code>	Destroys an element's alert.

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

事件

Bootstrap 警报插件提供额外的事件，可以直接取得调用方法和函式。

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

Copy

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

徽章 (Badge)

`.badge` 徽章样式的使用、数字提示扩展样式以及小规格徽章的示例和使用文档。

示例

`.badge` 可以嵌在标题中，并通过标题样式来适配其元素大小，因为其本身是通过相对字体大小和 `em` 单位的，所以有良好的弹性。

Example heading **New**

Example heading **New**

Example heading **New**

Example heading **New**

Example heading **New**

Example heading **New**

Copy

```
<h1>Example heading <span class="badge badge-secondary">New</span></h1>  
<h2>Example heading <span class="badge badge-secondary">New</span></h2>  
<h3>Example heading <span class="badge badge-secondary">New</span></h3>  
<h4>Example heading <span class="badge badge-secondary">New</span></h4>  
<h5>Example heading <span class="badge badge-secondary">New</span></h5>
```

```
<h6>Example heading <span class="badge badge-secondary">New</span></h6>
```

徽章可用作链接或按钮的一部分，以提供统计数字样式。

Notifications 4

```
<button type="button" class="btn btn-primary">
  Notifications <span class="badge badge-light">4</span>
</button>
```

Note that depending on how they are used, badges may be confusing for users of screen readers and similar assistive technologies. While the styling of badges provides a visual cue as to their purpose, these users will simply be presented with the content of the badge. Depending on the specific situation, these badges may seem like random additional words or numbers at the end of a sentence, link, or button.

Unless the context is clear (as with the “Notifications” example, where it is understood that the “4” is the number of notifications), consider including additional context with a visually hidden piece of additional text.

Profile 9unread messages

```
<button type="button" class="btn btn-primary">
  Profile <span class="badge badge-light">9</span>
  <span class="sr-only">unread messages</span>
</button>
```

情景变化

加入以下的 Class 样式来定义 `.badge` 的变化（颜色、大小等-译者注）：

Primary Secondary Success Danger Warning Info Light Dark

Copy

```
<span class="badge badge-primary">Primary</span>
<span class="badge badge-secondary">Secondary</span>
<span class="badge badge-success">Success</span>
<span class="badge badge-danger">Danger</span>
<span class="badge badge-warning">Warning</span>
<span class="badge badge-info">Info</span>
<span class="badge badge-light">Light</span>
<span class="badge badge-dark">Dark</span>
```

网页中传达辅助技术及其背后的意义

使用颜色添加意义只提供一个视觉指示，这不会传达给需要辅助技术（如盲人、听力障碍者）的用户，也就决定了诸如屏幕阅读器并不能读出颜色本身的意义。一般推荐确保由颜色表示的信息从内容本身（例如可见文本）中显而易见，或者通过替代方法，例如隐藏在 `.sr-only` 该类中的附加文本来创造更多的辅助传达技术。

`.sr-only`，全称 *screen reader only*，意为：(仅供)屏幕阅读器，这个 class 主要用于增强 accessibility(可访问性)，有时候 UI 上会出现一些仅供视觉识别的元素，比如说“菜单按钮”，只有视力正常的人才能清楚辨识这些元素的作用。而残障人士，比如弱势或盲人是不可可能知道这些视觉识别元素是什么的。他们上网使用的是屏幕阅读器，也就是 screen reader (sr)，屏幕阅读器需要找到能辨识的文本说明然后“读”出来给用户听。问题是图形元素怎么可能“读出来”呢？因此我们还要写上这些元素的文本说明，但是又不需要展示给普通用户看到，于是加上 `sr-only` 的意义就在于能保证屏幕阅读器正确读取且不会影响 UI 的视觉呈现（译者补充）。

椭圆形胶囊标签

使用 `.badge-pill` 样式，可以使标签更加圆润（具体有较大的 `border-radius` 边框半径和水平 `padding`），如果你错过了 V3 的标签这是有用的（这是 Bootstrap 4 中的特色功能）。

Primary Secondary Success Danger Warning Info Light Dark

Copy

```
<span class="badge badge-pill badge-primary">Primary</span>
<span class="badge badge-pill badge-secondary">Secondary</span>
<span class="badge badge-pill badge-success">Success</span>
<span class="badge badge-pill badge-danger">Danger</span>
<span class="badge badge-pill badge-warning">Warning</span>
<span class="badge badge-pill badge-info">Info</span>
<span class="badge badge-pill badge-light">Light</span>
<span class="badge badge-pill badge-dark">Dark</span>
```

链接它

`.badge-*` 也可以在 `<a>` 元素上使用，并实现悬停、焦点、状态等效果。

Primary Secondary Success Danger Warning Info Light Dark

Copy

```
<a href="#" class="badge badge-primary">Primary</a>
<a href="#" class="badge badge-secondary">Secondary</a>
<a href="#" class="badge badge-success">Success</a>
<a href="#" class="badge badge-danger">Danger</a>
<a href="#" class="badge badge-warning">Warning</a>
<a href="#" class="badge badge-info">Info</a>
<a href="#" class="badge badge-light">Light</a>
<a href="#" class="badge badge-dark">Dark</a>
```

面包屑导航(Breadcrumb)

在通过 Bootstrap 的内置 CSS 样式，自动添加分隔符、并呈现导航层次和网页结构结构，从而指示当前页面的位置为访客创造优秀用户体验。

概览

分隔符号是通过 `:before` 和 `content` 两个方法定义添加的。

```
<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item active" aria-current="page">Home</li>
  </ol>
</nav>
```

```
<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item"><a href="#">Home</a></li>
```

```
    <li class="breadcrumb-item active" aria-current="page">Library</li>
  </ol>
</nav>

<nav aria-label="breadcrumb">
  <ol class="breadcrumb">
    <li class="breadcrumb-item"><a href="#">Home</a></li>
    <li class="breadcrumb-item"><a href="#">Library</a></li>
    <li class="breadcrumb-item active" aria-current="page">Data</li>
  </ol>
</nav>
```

无障碍处理

为了为障碍浏览提供方便，针对面包屑这样具备导航功能的模块，建议添加一个有意义的标签即 `aria-label="breadcrumb"` 来描述 `<nav>` 元素，以及使用 `aria-current="page"` 到这组导航的最后一个项目，以标明当前页面名称（路径）

更多信息，敬请参阅 [WAI-ARIA Authoring Practices for the breadcrumb pattern](#)（面包屑导航的无障碍研究）。

按钮(Button)

使用 Bootstrap 的自定义按钮样式，并广泛用于表单、对话框等场景中的操作，并支持多种大小、状态等一系列变量定义。

示例

Bootstrap 包括多个预定义的按钮样式，每个样式都有自己的语义目的，另外还有一些额外的功能可以用于更多的控制。



```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>

<button type="button" class="btn btn-link">Link</button>
```

网页中传达辅助技术及其背后的意义

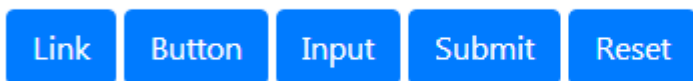
使用颜色添加意义只提供一个视觉指示，这不会传达给需要辅助技术（如盲人、听力障碍者）的用户，也就决定了诸如屏幕阅读器并不能读出颜色本身的意义。一般推荐确保由颜色表示的信息从内容本身（例如可见文本）中显而易见，或者通过替代方法，例如隐藏在 `.sr-only` 该类中的附加文本来创造更多的辅助传达技术。

`.sr-only`，全称 *screen reader only*，意为：(仅供)屏幕阅读器，这个 class 主要用于增强 accessibility(可访问性)，有时候 UI 上会出现一些仅供视觉识别的元素，比如说“菜单按钮”，只有视力正常的人才能清楚辨识这些元素的作用。而残障人士，比如弱势或盲人是不可能知道这些视觉识别元素是什么的。他们上网使用的是屏幕阅读器，也就是 screen reader (sr)，屏幕阅读器需要找到能辨识的文本说明然后“读”出来给用户听。问题是图形元素怎么可能“读出来”呢？因此我们还要写上这些元素的文本说明，但是又不需要展示给普通用户看到，于是加上 `sr-only` 的意义就在于能保证屏幕阅读器正确读取且不会影响 UI 的视觉呈现（译者补充）。

按钮标签

`.btn` 可以在 `<button>` 元素上使用，您也可以在 `<a>`、或 `<input>` 元素上使用这些 Class，同样能带来按钮效果（在少数浏览器下会有不同的渲染变异）。

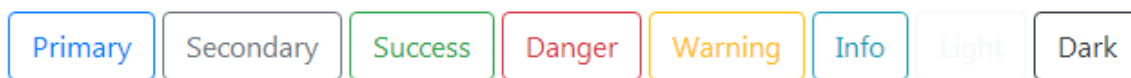
当在 `<a>` 上使用按钮元素，用于触发页内功能的元素（如折叠内容）上使用按钮类时，而不是链接到当前页面中的新页面或部分，应该给这些链接 `role="button"` 适当地传达其辅助技术的目的，从而友好的支持屏幕阅读器。



```
<a class="btn btn-primary" href="#" role="button">Link</a>
<button class="btn btn-primary" type="submit">Button</button>
<input class="btn btn-primary" type="button" value="Input">
<input class="btn btn-primary" type="submit" value="Submit">
<input class="btn btn-primary" type="reset" value="Reset">
```

轮廓按钮

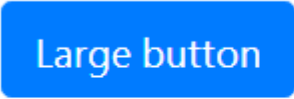
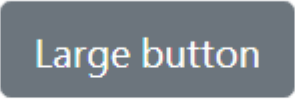
`.btn` 在引用中，如果需要一个按钮，但不需要带来的巨大的背景颜色（背景边框缩小）？用默认修饰符类替换 `.btn-outline-*` 任何按钮上的所有背景颜色和图像。



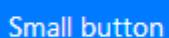
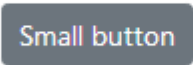
```
<button type="button" class="btn btn-outline-primary">Primary</button>
<button type="button" class="btn btn-outline-secondary">Secondary</button>
<button type="button" class="btn btn-outline-success">Success</button>
<button type="button" class="btn btn-outline-danger">Danger</button>
<button type="button" class="btn btn-outline-warning">Warning</button>
<button type="button" class="btn btn-outline-info">Info</button>
<button type="button" class="btn btn-outline-light">Light</button>
<button type="button" class="btn btn-outline-dark">Dark</button>
```

尺寸规格与大小定义

配合 `.btn-lg`、`.btn-sm` 两个邻近元素，可分别实现大规格按钮、小规格按钮的定义。

A large blue button with rounded corners and a subtle shadow, containing the text "Large button" in white.A large dark gray button with rounded corners and a subtle shadow, containing the text "Large button" in white.

```
<button type="button" class="btn btn-primary btn-lg">Large button</button>
<button type="button" class="btn btn-secondary btn-lg">Large button</button>
```

A small blue button with rounded corners and a subtle shadow, containing the text "Small button" in white.A small dark gray button with rounded corners and a subtle shadow, containing the text "Small button" in white.

```
<button type="button" class="btn btn-primary btn-sm">Small button</button>
<button type="button" class="btn btn-secondary btn-sm">Small button</button>
```


Create block level buttons—those that span the full width of a parent—by adding `.btn-block`.

A full-width blue button with rounded corners and a subtle shadow, containing the text "Block level button" in white.A full-width dark gray button with rounded corners and a subtle shadow, containing the text "Block level button" in white.

```
<button type="button" class="btn btn-primary btn-lg btn-block">Block level
button</button>
<button type="button" class="btn btn-secondary btn-lg btn-block">Block level
button</button>
```

启用状态

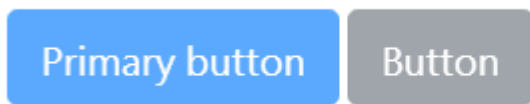
`.btn` 样式定义的按钮，默认就是启用状态（背景较深、边框较暗、带内阴影），如果你一定要使按钮固定为启用状态、不需要点击反馈，可以增加 `.active` 样式，并包括 `aria-pressed="true"` 属性，则状态显示为启用状态。

A large blue button with rounded corners and a subtle shadow, containing the text "Primary link" in white.A large dark gray button with rounded corners and a subtle shadow, containing the text "Link" in white.

```
<a href="#" class="btn btn-primary btn-lg active" role="button"
aria-pressed="true">Primary link</a>
<a href="#" class="btn btn-secondary btn-lg active" role="button"
aria-pressed="true">Link</a>
```

禁用状态

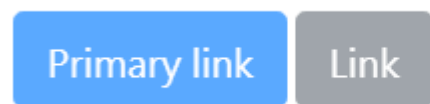
直接将 `disabled` 布尔属性添加到任何 `<button>` 元素（直接嵌套在 `html` 标签中，使按钮看起来处于非活动的禁用状态（点击不会有响应和弹性）。



```
<button type="button" class="btn btn-lg btn-primary" disabled>Primary
button</button>
<button type="button" class="btn btn-secondary btn-lg" disabled>Button</button>
```

使用 `<a>` 标签的禁用有所不同：

- `<a>` 标签不支持 `disabled` 属性，所以你必须增加 `.disabled` 属性，使之达到视觉禁用的效果。
- 未来，将包括更多的友好风格，以禁用按钮上的 `pointer-events` 属性，在支持该属性的浏览器中，会你看不到禁用的光标。
- 禁用按钮应包括 `aria-disabled="true"` 用于指示辅助技术元素的状态的属性。



```
<a href="#" class="btn btn-primary btn-lg disabled" role="button"
aria-disabled="true">Primary link</a>
<a href="#" class="btn btn-secondary btn-lg disabled" role="button"
aria-disabled="true">Link</a>
```

链接功能警告

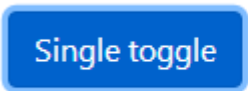
The `.disabled` class uses `pointer-events: none` to try to disable the link functionality of `<a>`s, but that CSS property is not yet standardized. In addition, even in browsers that do support `pointer-events: none`, keyboard navigation remains unaffected, meaning that sighted keyboard users and users of assistive technologies will still be able to activate these links. So to be safe, add a `tabindex="-1"` attribute on these links (to prevent them from receiving keyboard focus) and use custom JavaScript to disable their functionality.

按钮插件

按钮可以用在网页上的多种场合（做更多的组件），如工具栏、按钮组等。

切换状态

添加 `data-toggle="button"` 属性，可以切换按钮的 `active` 状态，如果你预先需要切换按钮，必须将 `.active` 样式、`aria-pressed="true"` 属性到 `<button>` 标签中。

Single toggle

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

Single toggle

```
</button>
```

复选框和单选框

Bootstrap 的 `.button` 样式也可以使用于其它元素，比如 `<label>` HTML 组件上，从而实现单选、复选效果。添加 `data-toggle="buttons"` 到 `.btn-group` 下的元素里，来启用它们的样式切换。

这些按钮的检查状态，只能通过 `click` 事件 进行更新，如果你使用其它方法更新输入，用 `<input type="reset">` 或手动应用输入 `checked` 属性，人为的在 `<label>` 上增加 `.active` 状态。

注意：预先选中的按钮需要你手动将 `.active` 定义上，在 `<label>` 中。

Checkbox 1 (pre-checked) Checkbox 2 Checkbox 3

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

Radio 1 (preselected) Radio 2 Radio 3

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

y

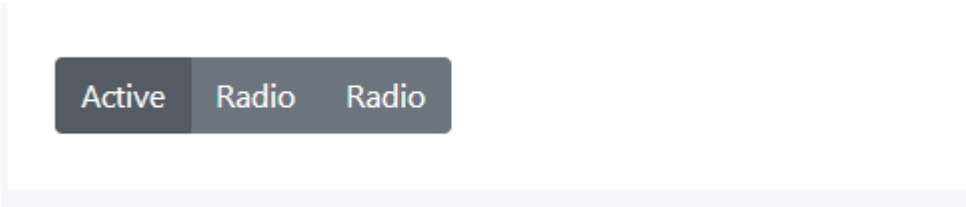
```
    <input type="radio" name="options" id="option3" autocomplete="off"> Radio 3
  </label>
</div>
```

上面的两个实例对应传统使用环境。

Bootstrap 4.0 正式版提供了全新的复选与单选 解决方案，其方案如下：



```
<div class="btn-group-toggle" data-toggle="buttons">
  <label class="btn btn-secondary active">
    <input type="checkbox" checked autocomplete="off"> Checked
  </label>
</div>
```



```
<div class="btn-group btn-group-toggle" data-toggle="buttons">
  <label class="btn btn-secondary active">
    <input type="radio" name="options" id="option1" autocomplete="off" checked>
Active
  </label>
  <label class="btn btn-secondary">
    <input type="radio" name="options" id="option2" autocomplete="off"> Radio
  </label>
  <label class="btn btn-secondary">
    <input type="radio" name="options" id="option3" autocomplete="off"> Radio
  </label>
</div>
```

方法

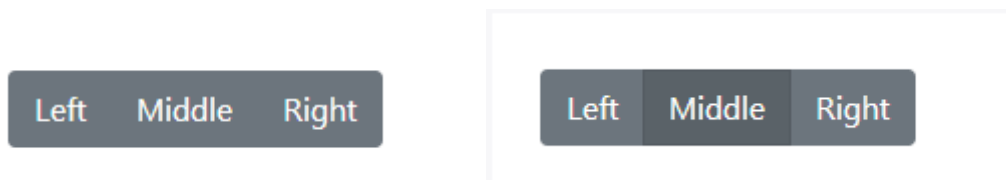
方法	属性
<code>\$(...).button('toggle')</code>	切换状态，给予按钮已经启用的外观。
<code>\$(...).button('dispose')</code>	销毁元素按钮的状态。

按钮组 (Btn-group)

使用按钮组合，可以把一系列按钮编组在一行里，并通过可选的 JavaScript 插件(方法)赋予按钮单选、复选等强化行为。

基本示例

将一系列的 `.btn` 包裹在 `.btn-group` 内，并使用[我们提供的插件](#)，可以实现选择按钮、选取块状区的行为功能。



```
<div class="btn-group" role="group" aria-label="Basic example">
  <button type="button" class="btn btn-secondary">Left</button>
  <button type="button" class="btn btn-secondary">Middle</button>
  <button type="button" class="btn btn-secondary">Right</button>
</div>
```

使用正确的标签并引用合法的 `role`

为了辅助技术（如屏幕阅读器）正确传达一系列按钮的分组，`role` 需要定义适当的属性，对于按钮组，引用的样式应该是 `role="group"`，如果是工具栏则应是 `role="toolbar"`。

此外，组和工具栏应该给予明确的标签，因为尽管存在正确的角色属性，大多数辅助设备并不能明确识别它们，上面实例我们使用了 `aria-label` 来定义，你也可以使用 `aria-labelledby` 方法定义。

按钮工具栏

根据需要使用样式定义，对按钮进行群组、间隔等定义，将按钮组的组合组合成为更复杂组件的按钮工具栏。



```
<div class="btn-toolbar" role="toolbar" aria-label="Toolbar with button groups">
  <div class="btn-group mr-2" role="group" aria-label="First group">
    <button type="button" class="btn btn-secondary">1</button>
    <button type="button" class="btn btn-secondary">2</button>
    <button type="button" class="btn btn-secondary">3</button>
    <button type="button" class="btn btn-secondary">4</button>
  </div>
  <div class="btn-group mr-2" role="group" aria-label="Second group">
    <button type="button" class="btn btn-secondary">5</button>
    <button type="button" class="btn btn-secondary">6</button>
    <button type="button" class="btn btn-secondary">7</button>
  </div>
  <button type="button" class="btn btn-secondary">8</button>
</div>
```



```

</div>
<div class="btn-group" role="group" aria-label="Third group">
  <button type="button" class="btn btn-secondary">8</button>
</div>
</div>

```

Feel free to mix input groups with button groups in your toolbars. Similar to the example above, you'll likely need some utilities though to space things properly.

1 2 3 4

@ Input group example

1 2 3 4

@ Input group example

```

<div class="btn-toolbar mb-3" role="toolbar" aria-label="Toolbar with button groups">
  <div class="btn-group mr-2" role="group" aria-label="First group">
    <button type="button" class="btn btn-secondary">1</button>
    <button type="button" class="btn btn-secondary">2</button>
    <button type="button" class="btn btn-secondary">3</button>
    <button type="button" class="btn btn-secondary">4</button>
  </div>
  <div class="input-group">
    <div class="input-group-prepend">
      <div class="input-group-text" id="btnGroupAddon">@</div>
    </div>
    <input type="text" class="form-control" placeholder="Input group example"
    aria-label="Input group example" aria-describedby="btnGroupAddon">
  </div>
</div>

```

```

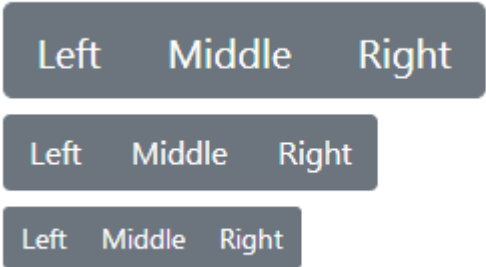
<div class="btn-toolbar justify-content-between" role="toolbar"
aria-label="Toolbar with button groups">
  <div class="btn-group" role="group" aria-label="First group">
    <button type="button" class="btn btn-secondary">1</button>
    <button type="button" class="btn btn-secondary">2</button>
    <button type="button" class="btn btn-secondary">3</button>
    <button type="button" class="btn btn-secondary">4</button>
  </div>
  <div class="input-group">
    <div class="input-group-prepend">
      <div class="input-group-text" id="btnGroupAddon2">@</div>
    </div>
    <input type="text" class="form-control" placeholder="Input group example"
    aria-label="Input group example" aria-describedby="btnGroupAddon2">
  </div>
</div>

```

```
</div>
</div>
```

大小规格和尺寸缩放

定义大小缩放，不需要将按钮中每个子元素都逐一定义，只需在 `.btn-group-*` 中增加 `.btn-group-*`，就能作用于组下的每个子按钮，实现样式缩放



```
<div class="btn-group btn-group-lg" role="group" aria-label="...">...</div>
<div class="btn-group" role="group" aria-label="...">...</div>
<div class="btn-group btn-group-sm" role="group" aria-label="...">...</div>
```

嵌套

将 `.btn-group` 放在另一个 `.btn-group` 里，可以实现按钮组与下拉菜单的组合。



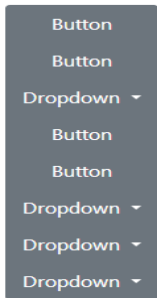
```
<div class="btn-group" role="group" aria-label="Button group with nested dropdown">
  <button type="button" class="btn btn-secondary">1</button>
  <button type="button" class="btn btn-secondary">2</button>

  <div class="btn-group" role="group">
    <button id="btnGroupDrop1" type="button" class="btn btn-secondary
dropdown-toggle" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
      Dropdown
    </button>
    <div class="dropdown-menu" aria-labelledby="btnGroupDrop1">
      <a class="dropdown-item" href="#">Dropdown link</a>
      <a class="dropdown-item" href="#">Dropdown link</a>
    </div>
  </div>
</div>
```

```
</div>
</div>
</div>
```

垂直排列

将一组按钮垂直排列，而不是水平排列，**不支持**分割式下拉菜单的定义。



```
<div class="btn-group-vertical">... </div>
```

.card 卡片组件（样式）

卡片是一个灵活的、可扩展的内容窗口，同时可以做出多种展示效果变体。

关于

.card 卡片组件是 Bootstrap 4 新增的一组重要样式，它是一个灵活的、可扩展的内容容器，包含了可选的卡片头和卡片脚、一个大范围的内容、上下文背景色以及强大的显示选项。

使用按钮组合，可以把一系列按钮编组在一行里，并通过可选的 JavaScript 插件(方法)赋予按钮单选、复选等强化行为。

如果你对 Bootstrap 3 很熟悉，卡片代替了我们旧的 panel、well 和 thumbnail 样式--那些组件类似的功能可以通过卡片的修饰类来实现。

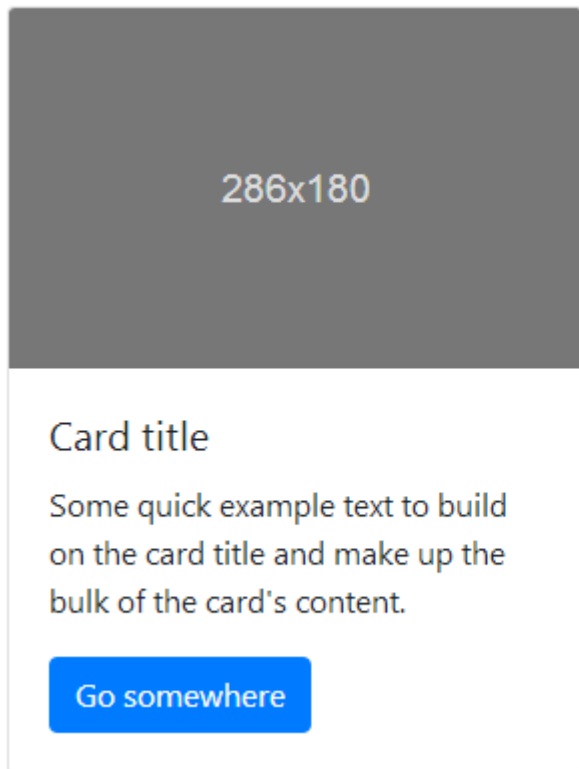
示例

卡片作为一个新式引入到 Bootstrap 4 中，可能样式、标记和扩展属性不会很多（未来会不断扩充发展），但它仍然可以提供许多控制项和定义方法，由于我们使用了 flex 弹性布局，使得它们可以轻松对齐、并方便的与其它 Bootstrap 组件混合搭配使用。

下面是一个具有混合内容并固定了宽度的基本 **.card** 卡片使用范例，如果 **.card** 卡片如果没有定义宽度，将自然地填满父元素的全宽-利用这个属性，我们可以轻松的订制各种尺寸的卡片。

Card title

Some quick example text to build on the card title and make up the bulk of the card's content.



```
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

内容类型

卡片支持多种多样的内容，包括图片、文本、列组、链接等，混合并匹配多种内容类型以创建你想要的卡片。

主体

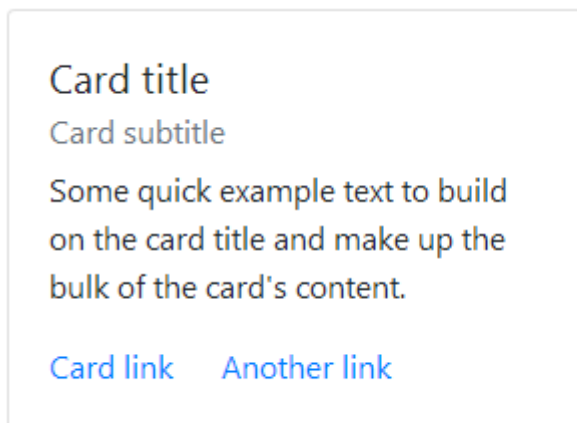
引用 `.card-body` 样式，可以建立起卡片的内容主体，如果你需要在一个 `.card` 中装置带边框的内容时，可以使用它。

```
<div class="card">
  <div class="card-body">
    This is some text within a card body.
  </div>
</div>
```

标题、文字和链接

通过 `.card-title` 和 `<h*>` 组合，可以添加卡片标题。同亲将 `.card-link` 与 `<a>` 结合使用，可以方便添加平行的链接。

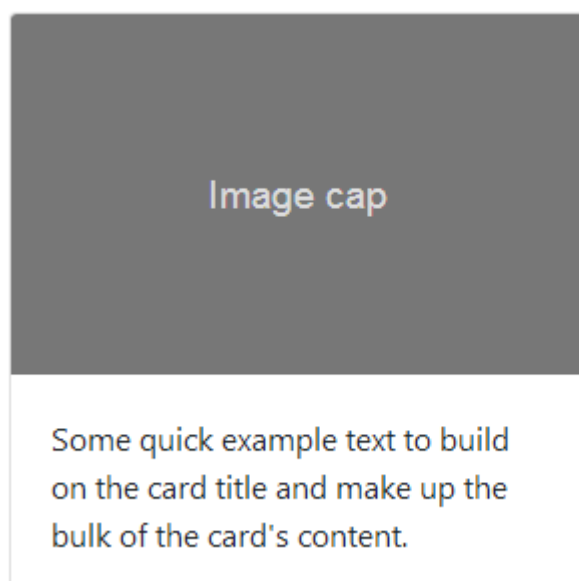
通过 `.card-subtitle` 和 `<h*>` 结合，可以添加副标题，如果 `.card-title` 和 `.card-subtitle` 组合放在 `.card-body` 中，则可对齐主、副标题。



```
<div class="card" style="width: 18rem;">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <h6 class="card-subtitle mb-2 text-muted">Card subtitle</h6>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
    <a href="#" class="card-link">Card link</a>
    <a href="#" class="card-link">Another link</a>
  </div>
</div>
```

图片

`.card-img-top` 定义一张图片在卡片的顶部，`.card-text` 定义文字在卡片中，当然你也可以在 `.card-text` 中设计自己的个性化 HTML 标签样式。



```
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
```

```
<div class="card-body">
  <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
</div>
</div>
```

列表组

建立一个包含内容的列表组卡片。

Cras justo odio
Dapibus ac facilisis in
Vestibulum at eros

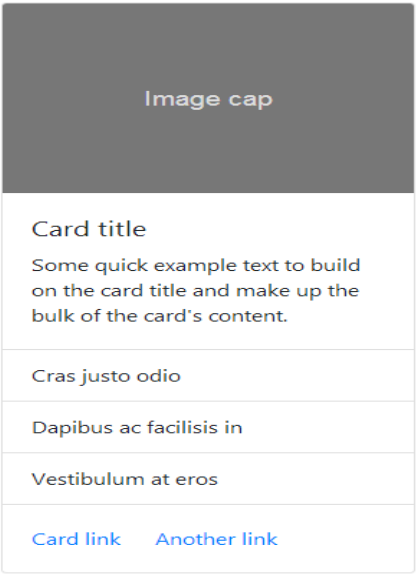
```
<div class="card" style="width: 18rem;">
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
</div>
```

Featured
Cras justo odio
Dapibus ac facilisis in
Vestibulum at eros

```
<div class="card" style="width: 18rem;">
  <div class="card-header">
    Featured
  </div>
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
</div>
```

混合样式

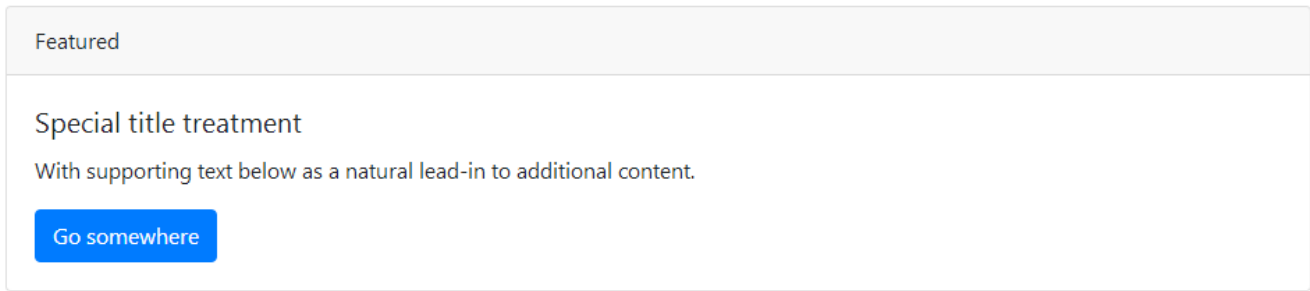
混合并结合多种内容形式来创建个性化卡片，下例即是将图像、块、文字以及列表整合在一个固定宽度的卡片中。
卡片标题



```
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
  <ul class="list-group list-group-flush">
    <li class="list-group-item">Cras justo odio</li>
    <li class="list-group-item">Dapibus ac facilisis in</li>
    <li class="list-group-item">Vestibulum at eros</li>
  </ul>
  <div class="card-body">
    <a href="#" class="card-link">Card link</a>
    <a href="#" class="card-link">Another link</a>
  </div>
</div>
```

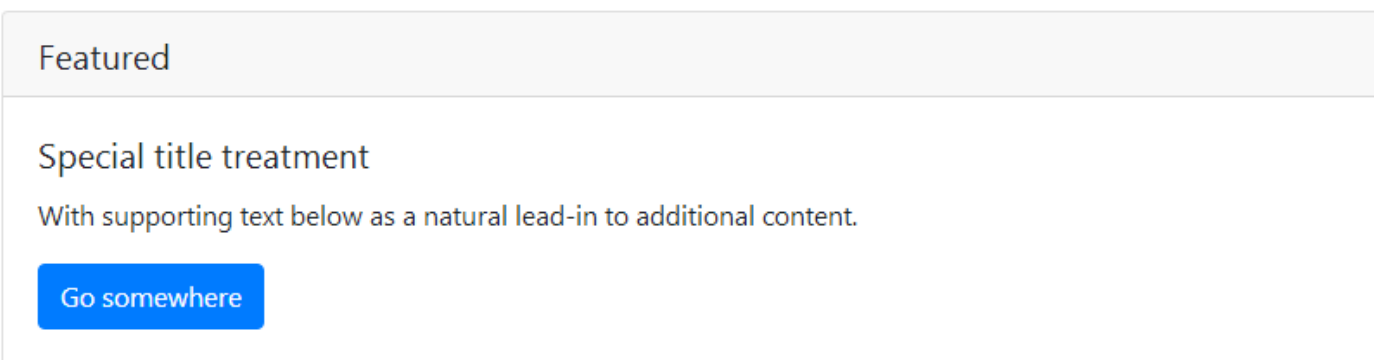
页眉页脚

在卡内添加可选的页眉和/或页脚。



```
<div class="card">
  <div class="card-header">
    Featured
  </div>
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

可以将 `<h*>` 元素添加到 `.card-header` 页头中。



```
<div class="card">
  <h5 class="card-header">Featured</h4>
  <div class="card-body">
    <h4 class="card-title">Special title treatment</h4>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
```


</div>

Quote

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere erat a ante.

— Someone famous in *Source Title*

```
<div class="card">
  <div class="card-header">
    Quote
  </div>
  <div class="card-body">
    <blockquote class="blockquote mb-0">
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere
erat a ante.</p>
      <footer class="blockquote-footer">Someone famous in <cite title="Source
Title">Source Title</cite></footer>
    </blockquote>
  </div>
</div>
```

Featured

Special title treatment

With supporting text below as a natural lead-in to additional content.

Go somewhere

2 days ago

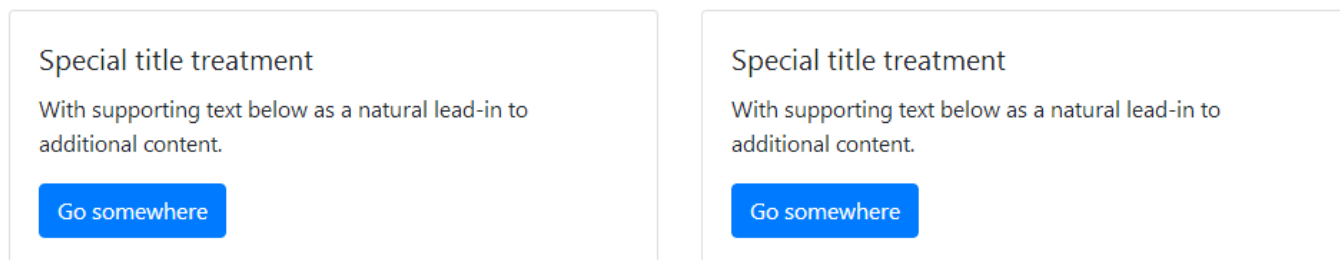
```
<div class="card text-center">
  <div class="card-header">
    Featured
  </div>
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
  <div class="card-footer text-muted">
    2 days ago
  </div>
</div>
```

缩放

卡片没有特定的宽度 **width** 定义，除非另有定义声明，否则它们的真实宽度将是 100%，您可以根据需要使用自定义 CSS，引入栅格系统、定义栅格系统 Sass mixins 或其它程式进行更改。

使用栅格系统

使用栅格系统，根据需求按行与列来装载卡片。



```
<div class="row">
  <div class="col-sm-6">
    <div class="card">
      <div class="card-body">
        <h5 class="card-title">Special title treatment</h5>
        <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
        <a href="#" class="btn btn-primary">Go somewhere</a>
      </div>
    </div>
  </div>
  <div class="col-sm-6">
    <div class="card">
      <div class="card-body">
        <h5 class="card-title">Special title treatment</h5>
        <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
        <a href="#" class="btn btn-primary">Go somewhere</a>
      </div>
    </div>
  </div>
</div>
```

使用通用全局属性

使用我们提供的[通用全局属性](#)来定义卡片的宽度。

Card title

With supporting text below as a natural lead-in to additional content.

Button

Card title

With supporting text below as a natural lead-in to additional content.

Button

```
<div class="card w-75">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Button</a>
  </div>
</div>
```

```
<div class="card w-50">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Button</a>
  </div>
</div>
```

自定义 CSS

在样式表中使用自定义 CSS 或使用内联样式设置宽度。

Special title treatment

With supporting text below as a natural lead-in to additional content.

Go somewhere

```
<div class="card" style="width: 18rem;">
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

文本对齐

使用我们的[文本对齐类](#)来更改或特定部份的文本对齐。

Special title treatment

With supporting text below as a natural lead-in to additional content.

[Go somewhere](#)

Special title treatment

With supporting text below as a natural lead-in to additional content.

[Go somewhere](#)

Special title treatment

With supporting text below as a natural lead-in to additional content.

[Go somewhere](#)

```
<div class="card" style="width: 18rem;">
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

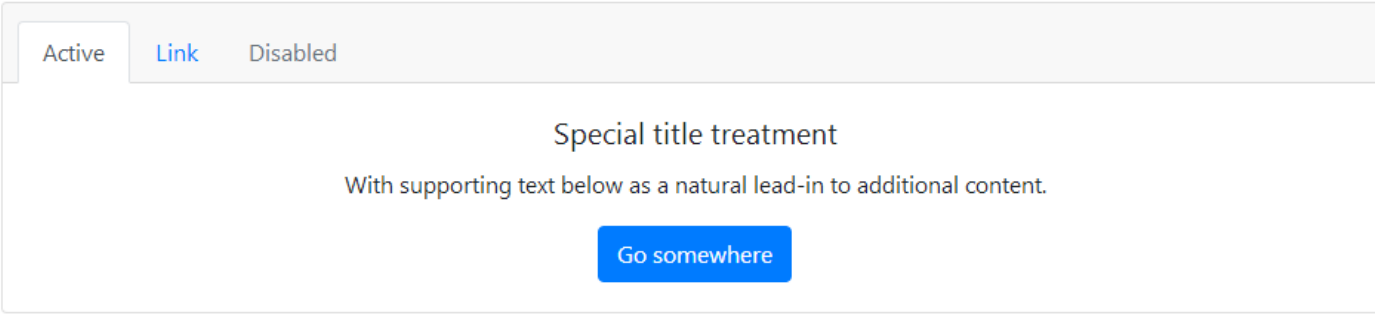
```
<div class="card text-center" style="width: 18rem;">
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

```
<div class="card text-right" style="width: 18rem;">
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
```

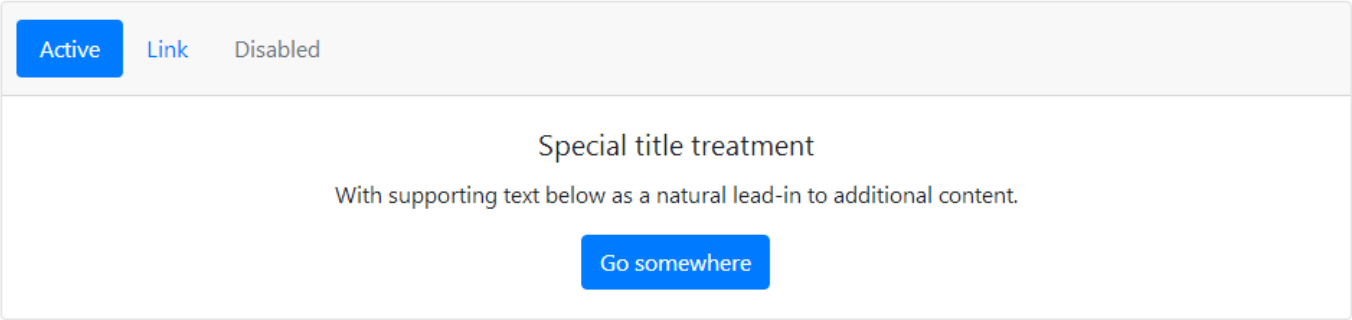
```
</div>
</div>
```

导航

使用 Bootstrap [导航组件](#)将导航元件添加到卡片的标题或块中



```
<div class="card text-center">
  <div class="card-header">
    <ul class="nav nav-tabs card-header-tabs">
      <li class="nav-item">
        <a class="nav-link active" href="#">Active</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
      </li>
      <li class="nav-item">
        <a class="nav-link disabled" href="#">Disabled</a>
      </li>
    </ul>
  </div>
  <div class="card-body">
    <h5 class="card-title">Special title treatment</h5>
    <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```



```
<div class="card text-center">
  <div class="card-header">
```

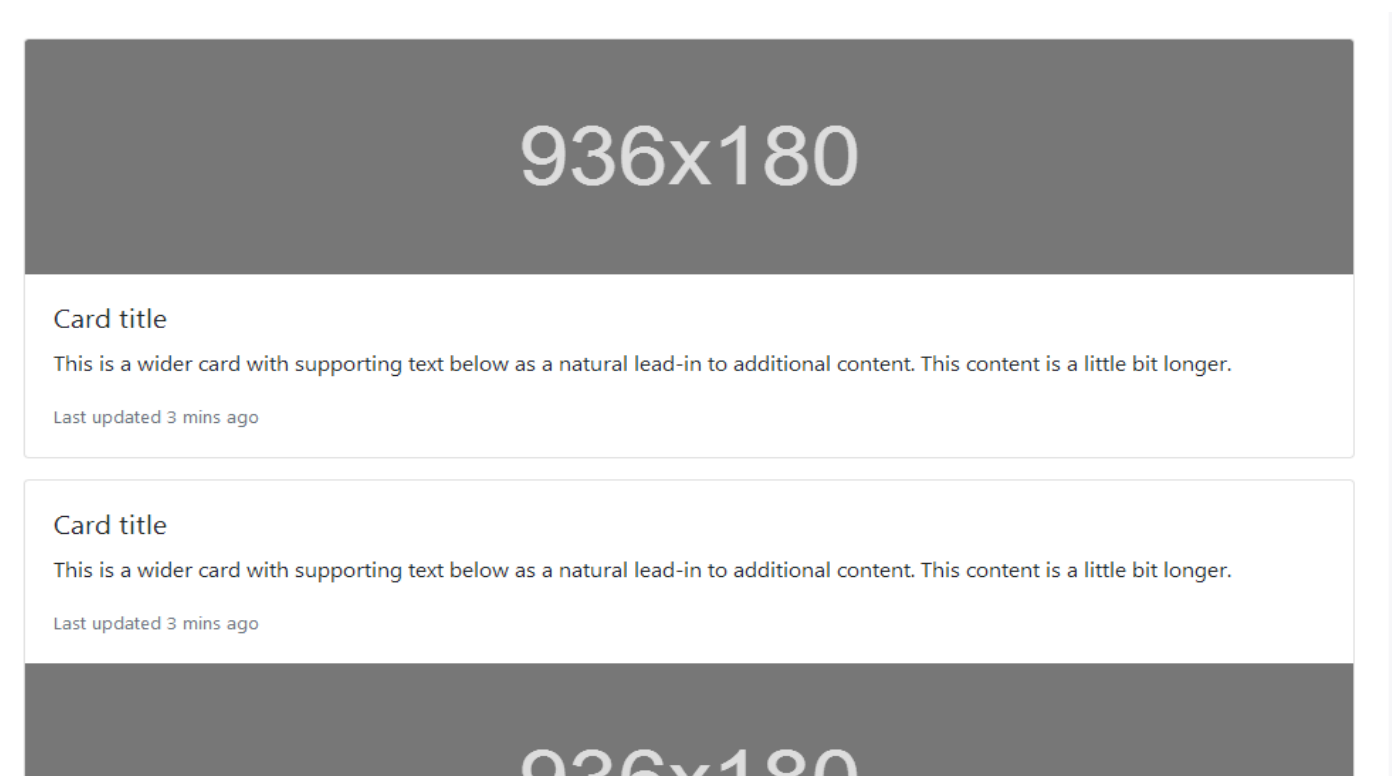
```
<ul class="nav nav-pills card-header-pills">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
</div>
<div class="card-body">
  <h5 class="card-title">Special title treatment</h5>
  <p class="card-text">With supporting text below as a natural lead-in to
additional content.</p>
  <a href="#" class="btn btn-primary">Go somewhere</a>
</div>
</div>
```

图片

卡片中包含一些选项来搭配图像，选择在卡片的任何一端附加 `.cad-img-*`，用卡片内容覆盖图像（如同背景），或者只是将图像置入到卡片当中。

图片覆盖

可以使用 `Image caps` 在卡片顶部或是底部，定义图片，如同页眉页脚效果。



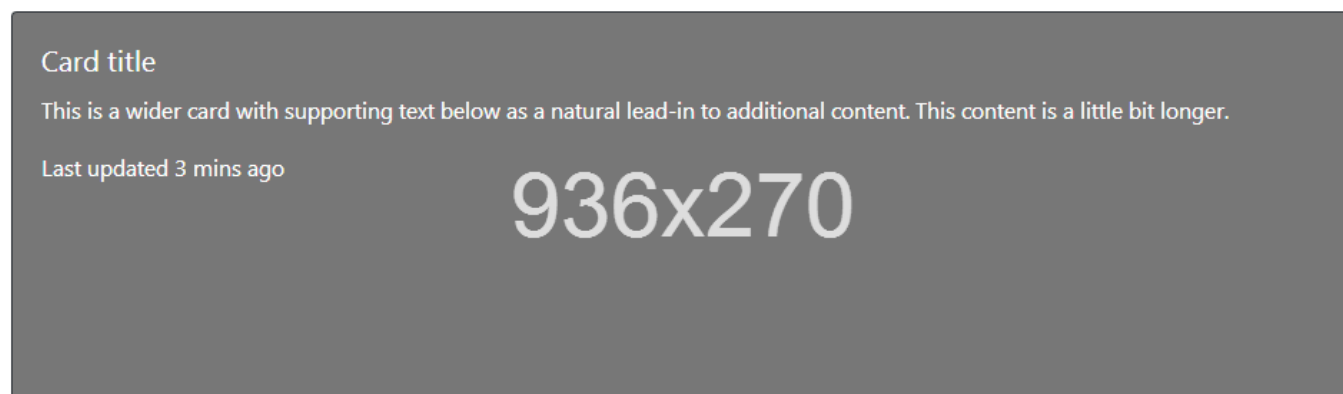
```

<div class="card mb-3">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">This is a wider card with supporting text below as a natural
lead-in to additional content. This content is a little bit longer.</p>
    <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
  </div>
</div>
<div class="card">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">This is a wider card with supporting text below as a natural
lead-in to additional content. This content is a little bit longer.</p>
    <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
  </div>
  
</div>

```

图像叠加覆盖

将图像转换为卡片背景，并覆盖卡片的文字。根据图像，你可以选择是否需要额外的样式或其它属性定义。



```

<div class="card bg-dark text-white">
  
  <div class="card-img-overlay">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">This is a wider card with supporting text below as a natural
lead-in to additional content. This content is a little bit longer.</p>
    <p class="card-text">Last updated 3 mins ago</p>
  </div>
</div>

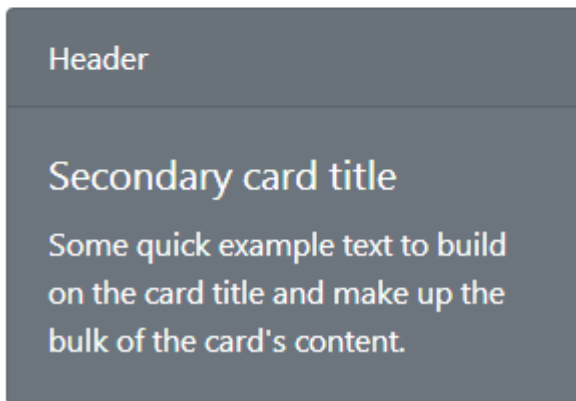
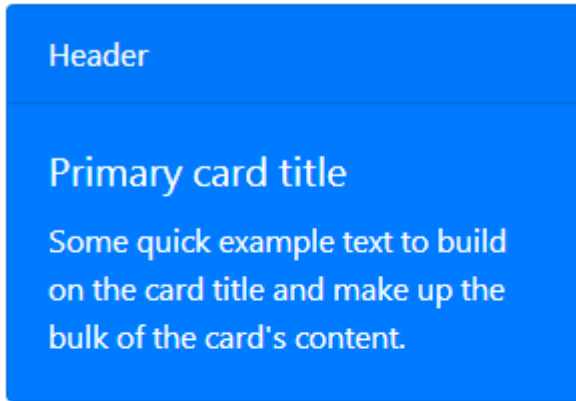
```

卡片样式

订制卡片、边框 、颜色等方法。

背景和颜色

使用 [文字和通用背景定义](#) 定义卡片的显示颜色。



```
<div class="card text-white bg-primary mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Primary card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card text-white bg-secondary mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Secondary card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card text-white bg-success mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Success card title</h5>
```

```
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card text-white bg-danger mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Danger card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card text-white bg-warning mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Warning card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card text-white bg-info mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Info card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card bg-light mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Light card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card text-white bg-dark mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Dark card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
```

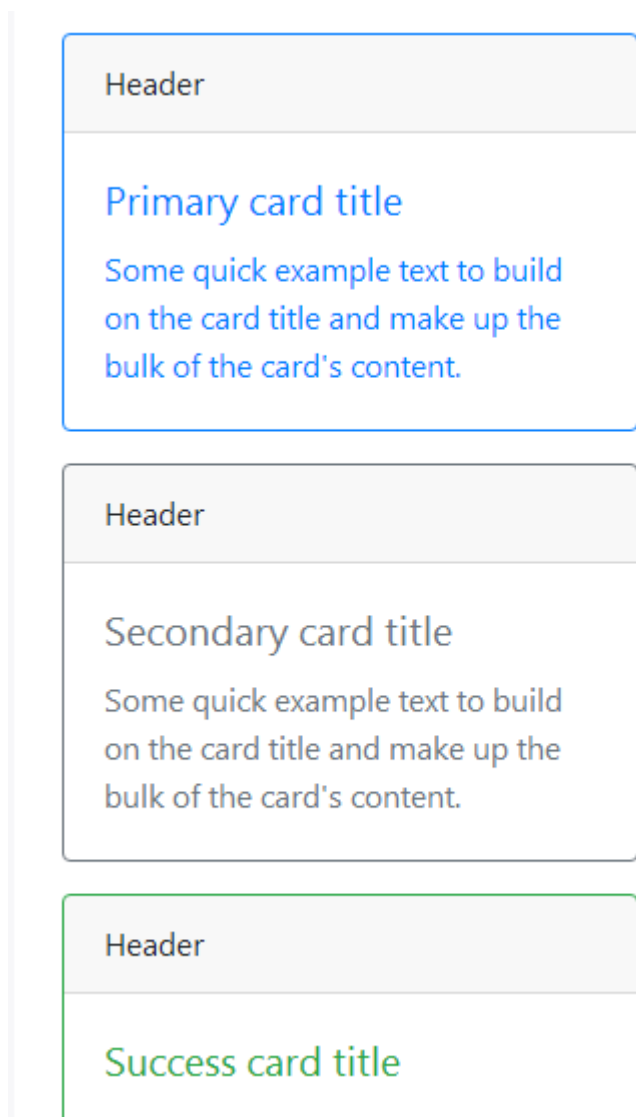
网页中传达辅助技术及其背后的意义

使用颜色添加意义只提供一个视觉指示，这不会传达给需要辅助技术（如盲人、听力障碍者）的用户，也就决定了诸如屏幕阅读器并不能读出颜色本身的意义。一般推荐确保由颜色表示的信息从内容本身（例如可见文本）中显而易见，或者通过替代方法，例如隐藏在 **.sr-only** 该类中的附加文本来创造更多的辅助传达技术。

.sr-only，全称 *screen reader only*,意为:(仅供)屏幕阅读器,这个 class 主要用于增强 accessibility(可访问性),有时候 UI 上会出现一些仅供视觉识别的元素，比如说“菜单按钮”，只有视力正常的人才能清楚辨识这些元素的作用。而残障人士，比如弱势或盲人是可能不知道这些视觉识别元素是什么的。他们上网使用的是屏幕阅读器，也就是 screen reader (sr)，屏幕阅读器需要找到能辨识的文本说明然后“读”出来给用户听。问题是图形元素怎么可能“读出来”呢？因此我们还要写上这些元素的文本说明，但是又不需要展示给普通用户看到，于是加上 sr-only 的意义就在于能保证屏幕阅读器正确读取且不会影响 UI 的视觉呈现（译者补充）。

边框

使用 [边框通用样式](#) 来改变卡片的 **border-color**、**.text-{color}**，或者在父层的 **.card** 上显示内容。



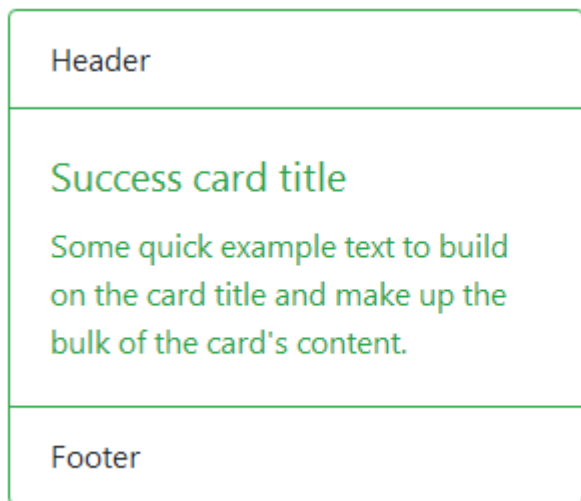
```
<div class="card border-primary mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-primary">
```

```
    <h5 class="card-title">Primary card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-secondary mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-secondary">
    <h5 class="card-title">Secondary card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-success mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-success">
    <h5 class="card-title">Success card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-danger mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-danger">
    <h5 class="card-title">Danger card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-warning mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-warning">
    <h5 class="card-title">Warning card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-info mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-info">
    <h5 class="card-title">Info card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-light mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body">
    <h5 class="card-title">Light card title</h5>
```

```
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
<div class="card border-dark mb-3" style="max-width: 20rem;">
  <div class="card-header">Header</div>
  <div class="card-body text-dark">
    <h5 class="card-title">Dark card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
</div>
```

Mixins 实用程序

您还可以更改卡上的页眉和页脚所需的边框，也能使用 `.bg-transparent` 删除其 `background-color` 背景颜色。



```
<div class="card border-success mb-3" style="max-width: 20rem;">
  <div class="card-header bg-transparent border-success">Header</div>
  <div class="card-body text-success">
    <h5 class="card-title">Success card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make
up the bulk of the card's content.</p>
  </div>
  <div class="card-footer bg-transparent border-success">Footer</div>
</div>
```

卡片排版

Bootstrap 除了对卡片内的内容可以进行设计排版外，还包括一系列布置选项，目前这些布置选项还不支持响应式。

卡片组

将多个卡片结为一个群组，使用他们具有相同的宽度和高度列。卡片组使用 `display: flex;` 来实现统一的布局。

311x180	311x180	311x180
<p>Card title</p> <p>This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.</p> <p>Last updated 3 mins ago</p>	<p>Card title</p> <p>This card has supporting text below as a natural lead-in to additional content.</p> <p>Last updated 3 mins ago</p>	<p>Card title</p> <p>This is a wider card with supporting text below as a natural lead-in to additional content. This card has even longer content than the first to show that equal height action.</p> <p>Last updated 3 mins ago</p>

```

<div class="card-group">
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This content is a little bit longer.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This card has supporting text below as a natural lead-in
to additional content.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This card has even longer content than the
first to show that equal height action.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>

```

</div>

当使用带页脚的卡片图时，他们的内容会自动水平对齐和栅格式布局。

311x180	311x180	311x180
<div>Card title</div> <div>This is a wider card with supporting text below as a natural lead-in to additional content. This content is a little bit longer.</div>	<div>Card title</div> <div>This card has supporting text below as a natural lead-in to additional content.</div>	<div>Card title</div> <div>This is a wider card with supporting text below as a natural lead-in to additional content. This card has even longer content than the first to show that equal height action.</div>
<div>Last updated 3 mins ago</div>	<div>Last updated 3 mins ago</div>	<div>Last updated 3 mins ago</div>

```
<div class="card-group">
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This content is a little bit longer.</p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This card has supporting text below as a natural lead-in
to additional content.</p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This card has even longer content than the
first to show that equal height action.</p>
    </div>
```

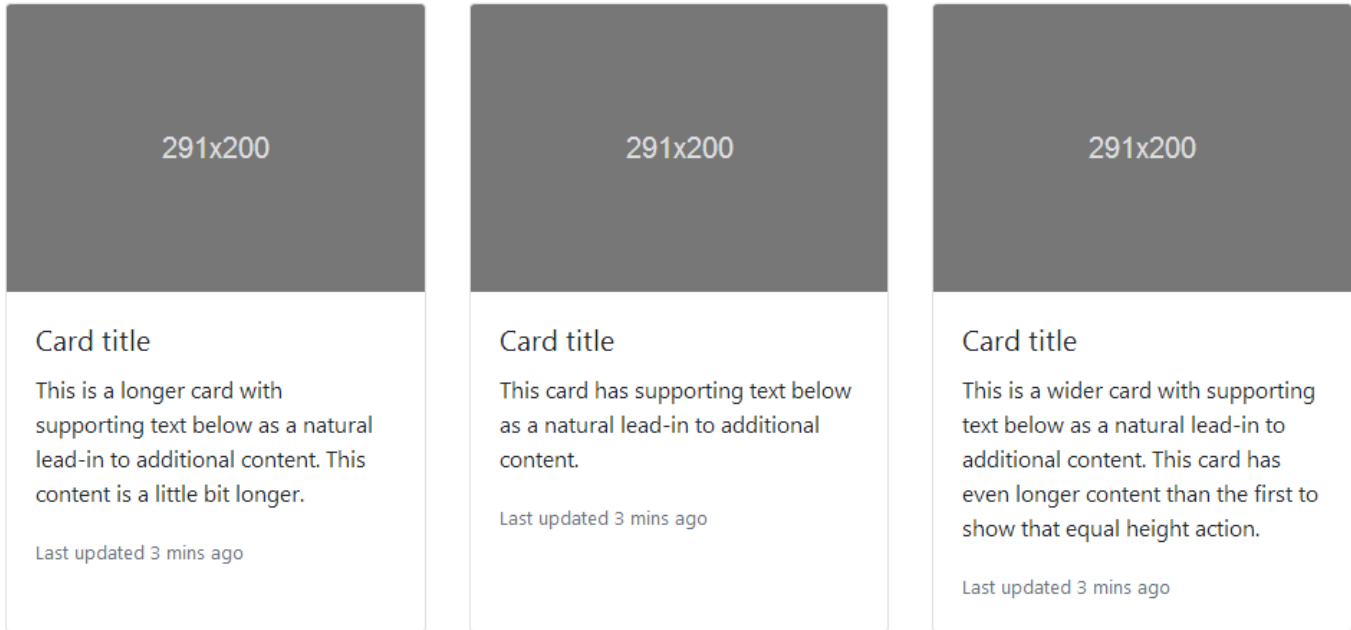
```

<div class="card-footer">
  <small class="text-muted">Last updated 3 mins ago</small>
</div>
</div>
</div>

```

Card decks 卡片阵列

需要一套相互不相连，但宽度和高度相同的卡片？使用卡片阵列（Card decks）吧。



```

<div class="card-deck">
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This is a longer card with supporting text below as a
natural lead-in to additional content. This content is a little bit longer.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This card has supporting text below as a natural lead-in
to additional content.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>
  <div class="card">
    

```

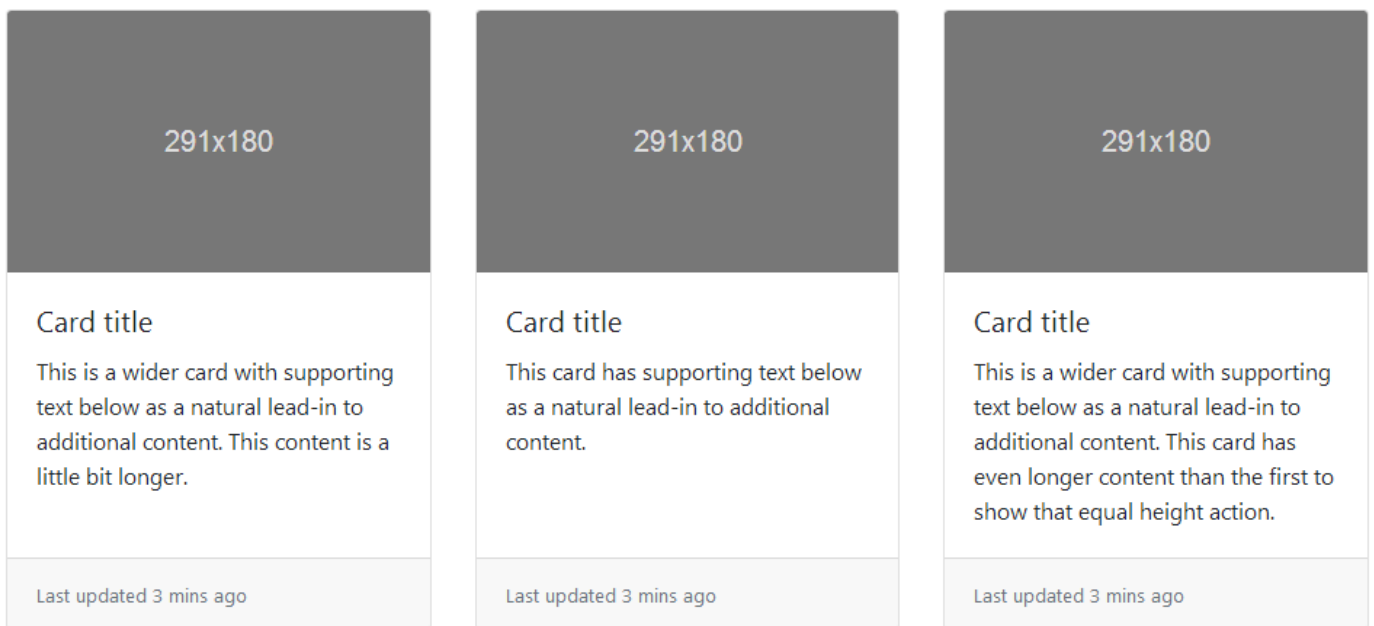


```

<div class="card-body">
  <h5 class="card-title">Card title</h5>
  <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This card has even longer content than the
first to show that equal height action.</p>
  <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
</div>
</div>
</div>

```

与卡片组一样，卡片阵列中的的的卡片页脚会自动排列。



```

<div class="card-deck">
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This content is a little bit longer.</p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This card has supporting text below as a natural lead-in
to additional content.</p>
    </div>
    <div class="card-footer">
      <small class="text-muted">Last updated 3 mins ago</small>
    </div>
  </div>

```

```

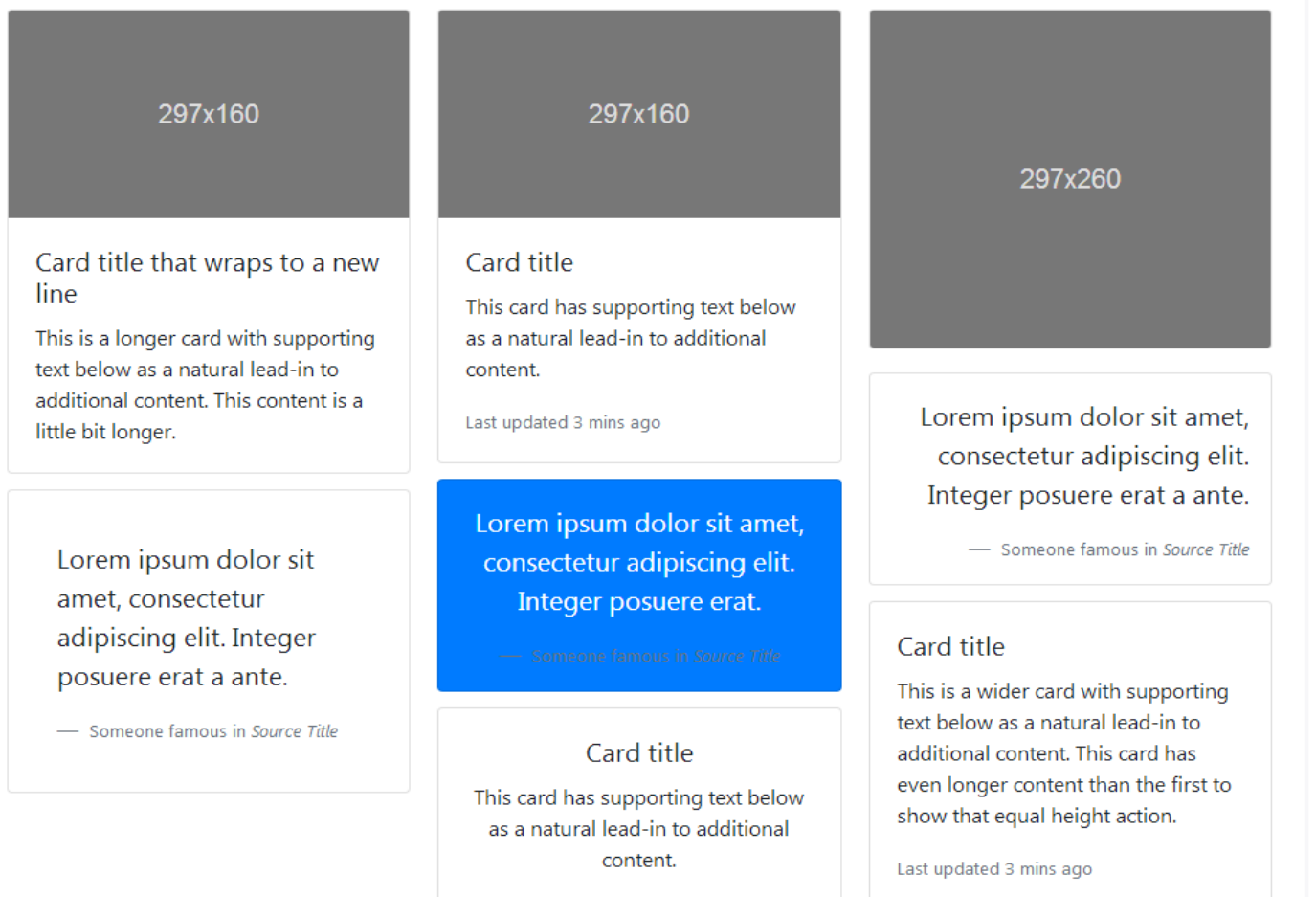
    </div>
</div>
<div class="card">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This card has even longer content than the
first to show that equal height action.</p>
  </div>
  <div class="card-footer">
    <small class="text-muted">Last updated 3 mins ago</small>
  </div>
</div>
</div>

```

多列卡片浮动排版

將卡片包在 `.card-columns` 中，可以將做出象 [Masonry](#) 网站的瀑布式排列卡片效果，卡片是使用 `column` 属性，而不是基于 `flexbox` 弹性布局，从而实现更方便实用的浮动对齐，顺序是从上到下、从左到右。

注意： 为了防止卡片排列突出栏目，我们必须为它们设置为 `display: inline-block`（当 `column-break-inside: avoid` 这个解决方案还没有生效时。



```

<div class="card-columns">
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title that wraps to a new line</h5>
      <p class="card-text">This is a longer card with supporting text below as a
natural lead-in to additional content. This content is a little bit longer.</p>
    </div>
  </div>
  <div class="card p-3">
    <blockquote class="blockquote mb-0 card-body">
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere
erat a ante.</p>
      <footer class="blockquote-footer">
        <small class="text-muted">
          Someone famous in <cite title="Source Title">Source Title</cite>
        </small>
      </footer>
    </blockquote>
  </div>
  <div class="card">
    
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This card has supporting text below as a natural lead-in
to additional content.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>
  <div class="card bg-primary text-white text-center p-3">
    <blockquote class="blockquote mb-0">
      <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere
erat.</p>
      <footer class="blockquote-footer">
        <small>
          Someone famous in <cite title="Source Title">Source Title</cite>
        </small>
      </footer>
    </blockquote>
  </div>
  <div class="card text-center">
    <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">This card has supporting text below as a natural lead-in
to additional content.</p>
      <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
    </div>
  </div>

```

```

</div>
<div class="card">
  
</div>
<div class="card p-3 text-right">
  <blockquote class="blockquote mb-0">
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer posuere
erat a ante.</p>
    <footer class="blockquote-footer">
      <small class="text-muted">
        Someone famous in <cite title="Source Title">Source Title</cite>
      </small>
    </footer>
  </blockquote>
</div>
<div class="card">
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">This is a wider card with supporting text below as a
natural lead-in to additional content. This card has even longer content than the
first to show that equal height action.</p>
    <p class="card-text"><small class="text-muted">Last updated 3 mins
ago</small></p>
  </div>
</div>
</div>

```

Card columns can also be extended and customized with some additional code. Shown below is an extension of the `.card-columns` class using the same CSS we use—CSS columns—to generate a set of responsive tiers for changing the number of columns.

Copy

```

.card-columns {
  @include media-breakpoint-only(lg) {
    column-count: 4;
  }
  @include media-breakpoint-only(xl) {
    column-count: 5;
  }
}

```

轮播效果(Carousel)

这是一个循环滚动的幻灯片组件，可以使用文本、图象水平不间断滚动，如同**旋转木马**一般。

工作原理

轮播效果是一个幻灯片效果，使用 CSS 3D 变形转换和一些 JavaScript 构建一内容循环播放，它适用于一系列图像、文本或自定义标记，还包括对上一个/下一个图的浏览控制和指令支持。

在支持 [Page Visibility API \(页面可见性\)](#) 的浏览器中，当网页对用户不可见时（如浏览器选项卡处于非活动状态、窗口最小化时），轮播效果控件会停止运动，从而节省性能。

轮播组件不支持互相嵌套-本身轮播大多不符合无障碍浏览的标准。

最后，如果你要自行编译构建 JS，记得 [需要 util.js](#)。

示例

轮播不带幻灯片尺寸标准化处理，因此你可能需要使用其它通用样式可自定义样式来调整其大小使之适当匹配。虽然轮播组件支持上一个/下一个控制和指令，但它们不是必备元素，可根据你的需要添加或自定义（展现不同的效果）。

通过 `.carousel` 命名样式引入轮播组件，同时为此控件设置唯的 ID-尤其是当你在同一页面使用多个轮播效果时这是必须的。

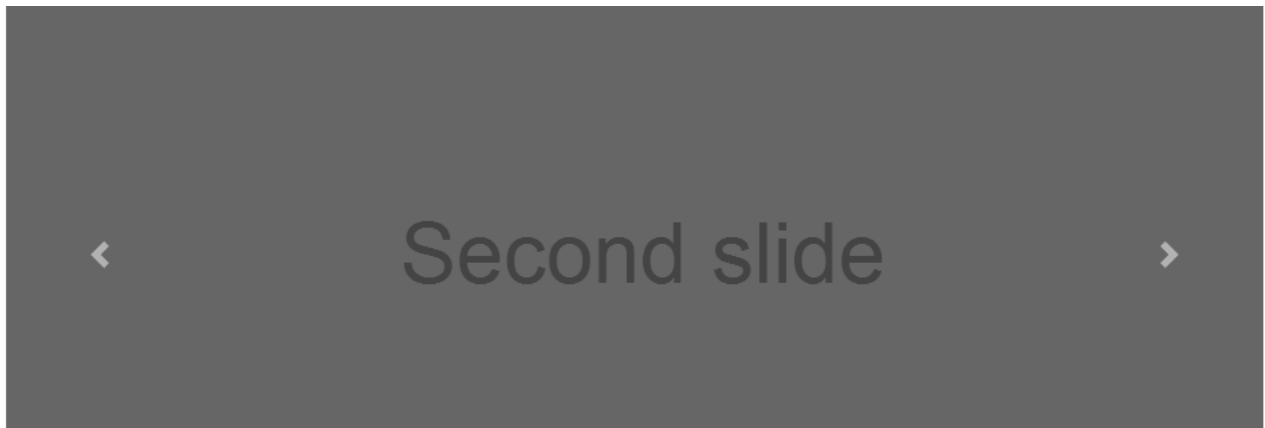
经典幻灯片效果

这是一个经典的幻灯片示例，请注意轮播上的图像引用了 `.d-block`、`.img-fluid` 两个样式，以修正浏览器预设的图像对齐带来的影响。

```
<div id="carouselExampleSlidesOnly" class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      
    </div>
    <div class="carousel-item">
      
    </div>
    <div class="carousel-item">
      
    </div>
  </div>
</div>
```

带控制器的效果

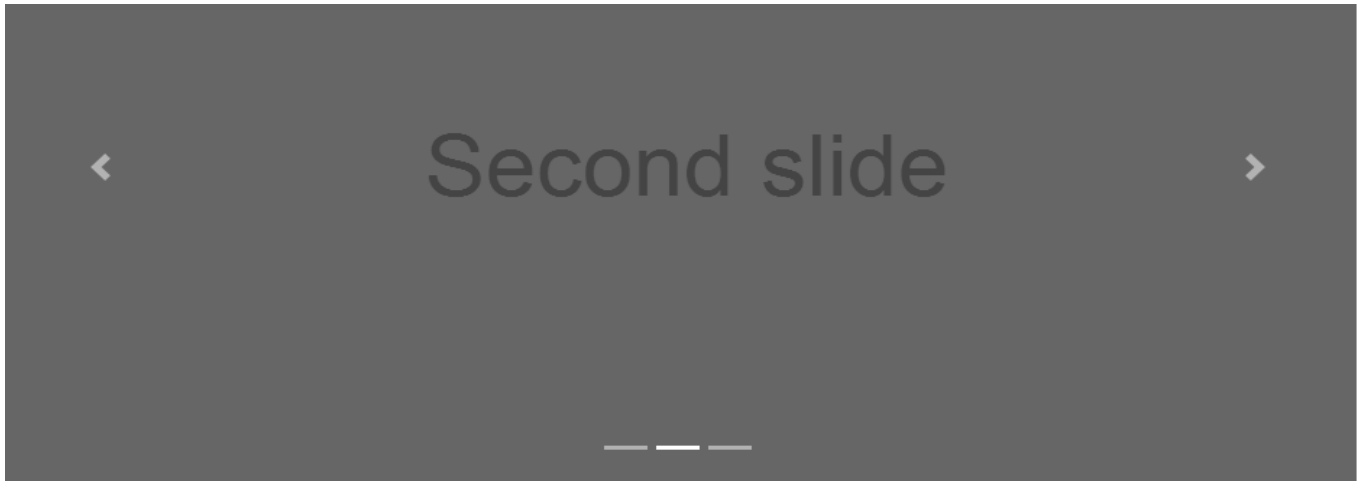
加上了上一个/下一个控制器:



```
<div id="carouselExampleControls" class="carousel slide" data-ride="carousel">
  <div class="carousel-inner">
    <div class="carousel-item active">
      
    </div>
    <div class="carousel-item">
      
    </div>
    <div class="carousel-item">
      
    </div>
  </div>
  <a class="carousel-control-prev" href="#carouselExampleControls" role="button"
data-slide="prev">
    <span class="carousel-control-prev-icon" aria-hidden="true"></span>
    <span class="sr-only">Previous</span>
  </a>
  <a class="carousel-control-next" href="#carouselExampleControls" role="button"
data-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="sr-only">Next</span>
  </a>
</div>
```

包含姿态指示器

也可以将当前所在幻灯片状态指示器添加到轮播效果控件中:



```
<div id="carouselExampleIndicators" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
    <li data-target="#carouselExampleIndicators" data-slide-to="0"
class="active"></li>
    <li data-target="#carouselExampleIndicators" data-slide-to="1"></li>
    <li data-target="#carouselExampleIndicators" data-slide-to="2"></li>
  </ol>
  <div class="carousel-inner">
    <div class="carousel-item active">
      
    </div>
    <div class="carousel-item">
      
    </div>
    <div class="carousel-item">
      
    </div>
  </div>
  <a class="carousel-control-prev" href="#carouselExampleIndicators"
role="button" data-slide="prev">
    <span class="carousel-control-prev-icon" aria-hidden="true"></span>
    <span class="sr-only">Previous</span>
  </a>
  <a class="carousel-control-next" href="#carouselExampleIndicators"
role="button" data-slide="next">
    <span class="carousel-control-next-icon" aria-hidden="true"></span>
    <span class="sr-only">Next</span>
  </a>
</div>
```

需要定义有效的初始状态元素

将 `.active` 样式添加到其中一个幻灯片（一般第一张），否则轮播效果将无法正常运行（展现）。

包含字幕的轮播

在 `.carousel-item` 中使用 `.carousel-caption` 添加字幕到您的轮播控件中，如果是输小的浏览器 viewport 上，会自动隐藏（隐藏文字呈现主图片轮播），引用的是 `.d-none` 定义，一旦到了中型 md 浏览设备或屏幕则调用 `.d-md-block` 样式使之呈现。



```
<div class="carousel-item">
  
  <div class="carousel-caption d-none d-md-block">
    <h3>...</h3>
    <p>...</p>
  </div>
</div>
```

用法

通过数据属性

使用数据属性可以轻松控制转盘的位置。`data-slide` 接受关键字，`prev` 或者 `next` 改变相对于其当前位置的滑动位置。或者，使用 `data-slide-to` 将原始幻灯片索引传递到 `data-slide-to="2"`，将幻灯片位置转换到以 0 开始的特定索引。

该 `data-ride="carousel"` 属性用于将轮播标记为从页面加载开始的动画，它不能与同一转盘的（冗余和不必要）显式 **JavaScript** 初始化结合使用。

通过 JavaScript

通过下面方法使用 JS 控制轮播：

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

选项

可以透過資料屬性或 JavaScript 調整選項。對於資料屬性，將選項名稱附加到 data-，如 data-interval=""。 可以通过数据属性或 JavaScript 调整（传递）选项，对于数据属性，选项名称追加到 data-，如 data-interval=""。

名称	Type 类型	默认值	描述
interval	number	5000	自动循环项目之间的延迟时间（即滚动时间），。果为 false，则传送带不会滚动。
keyboard	boolean	true	旋转木马是否应对键盘事件作出反应。
pause	string boolean	"hover"	悬停控制-注意移动平台的配置。If set to "hover", pauses the cycling of the carousel on mouseenter and resumes the cycling of the carousel on mouseleave. If set to false, hovering over the carousel won't pause it. On touch-enabled devices, when set to "hover", cycling will pause on touchend (once the user finished interacting with the carousel) for two intervals, before automatically resuming. Note that this is in addition to the above mouse behavior.
ride	string	false	在用户手动循环第一个项目后自动播放传送带， 如果"carousel"则加载时自动播放传送带。
wrap	boolean	true	转盘是否应该连续循环或难以停止。

方法

异步传输和转换

所有 API 都支持 异步传输 和 轮换。一旦转换事件发生（开始），直到 事件结束之前 不会结束。另外，在 过渡组件上的方法将被忽略。

请参阅[我们的 API 文档](#)了解更多。

.carousel(options)

通过 object 初始化，启动并执行轮播。

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

从左到右循环播放。

.carousel('pause')

通过事件停止幻灯片播放。

.carousel(number)

将轮播循环到特定的帧（基于 0，类似数组），在 目标被显示之前 回传给调用用者（即 slid.bs.carousel 事件之前）。

`.carousel('prev')`

将轮播指向前一帧幻灯片，**在前一个目标被显示之前**回传给调用者 (即 `slid.bs.carousel` 事件之前)。

`.carousel('next')`

将轮播指向后一帧幻灯片，**在前一个目标被显示之前**回传给调用者 (即 `slid.bs.carousel` 事件之前)。

`.carousel('dispose')`

销毁一个轮播的控件。

事件

Bootstrap 提供了两下事件给轮播控件使用，这两个事件都具有以下附加属性：

- `direction`: 轮播滚动的方向 ("`left`" 或 "`right`")。
- `relatedTarget`: 作为活动项目滑动到指定的 DOM 元素。
- `from`: 当前项目的索引。
- `to`: 下一个项目的索引。

所有的轮播事件都在轮播本身 (即 `<div class="carousel">`)下被触发。

事件类型 Event Type	描述
<code>slide.bs.carousel</code>	当用 <code>slide</code> 方法时，此事件会立即触发。
<code>slid.bs.carousel</code>	轮播完成切换后，此事件即被触发。

Copy

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

折叠面板 (Collapse)

Bootstrap 折叠板插件允许你在网页中用一点点 JavaScript 以及 CSS 类切换内容，控制内容的可见性。它是一个灵活的插件，使用少量的类（来自必需的过渡插件），以方便切换行为。

示例

点击下面任何一个按钮，通过类更改显示和隐藏另一个 class 包含的元素：

- `.collapse` 隐藏内容
- `.collapsing` 带动态效果的切换
- `.collapse.show` 显示内容

你可以使用带 `href` 属性的连接、或者带 `data-target` 属性的按钮来创建折叠效果-这两种情况下 `data-toggle="collapse"` 属性都是必须的。

Link with href

Button with data-target

Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad squid. Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea proident.

```
<p>
  <a class="btn btn-primary" data-toggle="collapse" href="#collapseExample"
role="button" aria-expanded="false" aria-controls="collapseExample">
    Link with href
  </a>
  <button class="btn btn-primary" type="button" data-toggle="collapse"
data-target="#collapseExample" aria-expanded="false"
aria-controls="collapseExample">
    Button with data-target
  </button>
</p>
<div class="collapse" id="collapseExample">
  <div class="card card-body">
    Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry
richardson ad squid. Nihil anim keffiyeh helvetica, craft beer labore wes anderson
cred nesciunt sapiente ea proident.
  </div>
</div>
```

多目标控制

可以在 `<button>` 或者 `<a>` 标签上，通过 JQuery 选择器来显示和隐藏多个元素（或者多个 `<button>`、`<a>` 元素来控制显示/隐藏一个元素），如果被引用对象的 `href` 或者 `data-target` 属性定义正确。

Toggle first element Toggle second element Toggle both elements

Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad squid. Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea proident.

Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry richardson ad squid. Nihil anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea proident.

```
<p>
  <a class="btn btn-primary" data-toggle="collapse" href="#multiCollapseExample1"
role="button" aria-expanded="false" aria-controls="multiCollapseExample1">Toggle
first element</a>
  <button class="btn btn-primary" type="button" data-toggle="collapse"
data-target="#multiCollapseExample2" aria-expanded="false"
aria-controls="multiCollapseExample2">Toggle second element</button>
  <button class="btn btn-primary" type="button" data-toggle="collapse"
data-target=".multi-collapse" aria-expanded="false"
aria-controls="multiCollapseExample1 multiCollapseExample2">Toggle both
elements</button>
</p>
<div class="row">
  <div class="col">
    <div class="collapse multi-collapse" id="multiCollapseExample1">
      <div class="card card-body">
        Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry
richardson ad squid. Nihil anim keffiyeh helvetica, craft beer labore wes anderson
cred nesciunt sapiente ea proident.
      </div>
    </div>
  </div>
  <div class="col">
    <div class="collapse multi-collapse" id="multiCollapseExample2">
      <div class="card card-body">
        Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry
richardson ad squid. Nihil anim keffiyeh helvetica, craft beer labore wes anderson
cred nesciunt sapiente ea proident.
      </div>
    </div>
  </div>
</div>
```

手风琴折叠范例

结合 [card](#) 卡片组件使用，可以扩展折叠组件为手风琴效果。

[Collapsible Group Item #1](#)

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

[Collapsible Group Item #2](#)

[Collapsible Group Item #3](#)

```
<div id="accordion">
  <div class="card">
    <div class="card-header" id="headingOne">
      <h5 class="mb-0">
        <button class="btn btn-link" data-toggle="collapse"
data-target="#collapseOne" aria-expanded="true" aria-controls="collapseOne">
          Collapsible Group Item #1
        </button>
      </h5>
    </div>

    <div id="collapseOne" class="collapse show" aria-labelledby="headingOne"
data-parent="#accordion">
      <div class="card-body">
        Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry
richardson ad squid. 3 wolf moon officia aute, non cupidatat skateboard dolor brunch.
Food truck quinoa nesciunt laborum eiusmod. Brunch 3 wolf moon tempor, sunt aliqua
put a bird on it squid single-origin coffee nulla assumenda shoreditch et. Nihil
anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea
proident. Ad vegan excepteur butcher vice lomo. Leggings occaecat craft beer
farm-to-table, raw denim aesthetic synth nesciunt you probably haven't heard of them
accusamus labore sustainable VHS.
      </div>
    </div>
  </div>
  <div class="card">
    <div class="card-header" id="headingTwo">
      <h5 class="mb-0">
        <button class="btn btn-link collapsed" data-toggle="collapse"
data-target="#collapseTwo" aria-expanded="false" aria-controls="collapseTwo">
          Collapsible Group Item #2
        </button>
      </h5>
    </div>
  </div>
```

```

<div id="collapseTwo" class="collapse" aria-labelledby="headingTwo"
data-parent="#accordion">
  <div class="card-body">
    Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry
richardson ad squid. 3 wolf moon officia aute, non cupidatat skateboard dolor brunch.
Food truck quinoa nesciunt laborum eiusmod. Brunch 3 wolf moon tempor, sunt aliqua
put a bird on it squid single-origin coffee nulla assumenda shoreditch et. Nihil
anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea
proident. Ad vegan excepteur butcher vice lomo. Leggings occaecat craft beer
farm-to-table, raw denim aesthetic synth nesciunt you probably haven't heard of them
accusamus labore sustainable VHS.
  </div>
</div>
</div>
<div class="card">
  <div class="card-header" id="headingThree">
    <h5 class="mb-0">
      <button class="btn btn-link collapsed" data-toggle="collapse"
data-target="#collapseThree" aria-expanded="false"
aria-controls="collapseThree">
        Collapsible Group Item #3
      </button>
    </h5>
  </div>
  <div id="collapseThree" class="collapse" aria-labelledby="headingThree"
data-parent="#accordion">
    <div class="card-body">
      Anim pariatur cliche reprehenderit, enim eiusmod high life accusamus terry
richardson ad squid. 3 wolf moon officia aute, non cupidatat skateboard dolor brunch.
Food truck quinoa nesciunt laborum eiusmod. Brunch 3 wolf moon tempor, sunt aliqua
put a bird on it squid single-origin coffee nulla assumenda shoreditch et. Nihil
anim keffiyeh helvetica, craft beer labore wes anderson cred nesciunt sapiente ea
proident. Ad vegan excepteur butcher vice lomo. Leggings occaecat craft beer
farm-to-table, raw denim aesthetic synth nesciunt you probably haven't heard of them
accusamus labore sustainable VHS.
    </div>
  </div>
</div>
</div>

```

你也可以使用自定义样式创建手风琴效果，只要添加 `data-children` 属性，并指定一组相邻元素来切换(如 `.item`)，然后使用与上述相同的属性和 `class`，来切换/隐藏其关联的内容。

[Toggle item](#)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed pretium lorem non vestibulum scelerisque. Proin a vestibulum sem, eget tristique massa. Aliquam lacinia rhoncus nibh quis ornare.

[Toggle item 2](#)

Copy

```

<div id="exampleAccordion" data-children=".item">

```

```

<div class="item">
  <a data-toggle="collapse" data-parent="#exampleAccordion"
href="#exampleAccordion1" aria-expanded="true"
aria-controls="exampleAccordion1">
    Toggle item
  </a>
  <div id="exampleAccordion1" class="collapse show" role="tabpanel">
    <p class="mb-3">
      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed pretium lorem
non vestibulum scelerisque. Proin a vestibulum sem, eget tristique massa. Aliquam
lacinia rhoncus nibh quis ornare.
    </p>
  </div>
</div>
<div class="item">
  <a data-toggle="collapse" data-parent="#exampleAccordion"
href="#exampleAccordion2" aria-expanded="false"
aria-controls="exampleAccordion2">
    Toggle item 2
  </a>
  <div id="exampleAccordion2" class="collapse" role="tabpanel">
    <p class="mb-3">
      Donec at ipsum dignissim, rutrum turpis scelerisque, tristique lectus.
Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis
egestas. Vivamus nec dui turpis. Orci varius natoque penatibus et magnis dis
parturient montes, nascetur ridiculus mus.
    </p>
  </div>
</div>
</div>

```

无障碍浏览提示(易用性)

建议添加 `aria-expanded` 到控件元素中，该属性能明确将与控件相关的可折叠元素之当前状态传达给屏幕阅读器和类似的辅助技术。如果折叠块元素默认是闭合的，它必须拥有一个 `aria-expanded="false"` 值。如果你用 `.show` 类设置则可以定义折叠控件为默认打开，在控制器上设置 `aria-expanded="true"` 即可。插件会根据折叠块元素是打开还是关闭的，自动切换这个属性（通过 JavaScript，或者因为用户触发的另一个控件元素也绑定到相同的折叠元素）。

此外，如果控件元素只对准一个单个元素——即 `data-target` 的值指向一个 id 选择器，你可以给这个控件元素添加额外的 `aria-controls` 属性，容纳这个折叠块元素的 id。现代的屏幕阅读器以及类似的辅助技术利用这个属性向用户提供额外的快捷方式，直接导航到折叠块元素本身。

用法

折叠插件使用一些 Class 类来处理复杂的事务：

- `.collapse` 隐藏内容
- `.collapse.show` 显示内容

- `.collapsing` 在转换开始时被添加，当它完成时则移除。

这些类可以在 `_transitions.scss` 文件中查阅。

利用 data 数据属性

将 `data-toggle="collapse"` 和 `data-target` 添加到元素中，可自动指定折叠面板的控制项，其中 `data-target` 属性接受 CSS 选择器，以应用折叠。确保向可折叠面板添加 `collapse` 添加到可折叠面板组件之上。如果你希望它默认是打开的，可定义额外的 `show` 属性。

为了给一个折叠块控件添加类似手风琴组的效果，还需要添加 `data` 属性 `data-parent="#selector"`。可以参考下面的演示了解实践例子。

利用 JavaScript

人为启用它:

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

选项

可通过 `data` 属性或 JavaScript 传递选项。如用 `data` 属性，请把选项名追加到 `data-` 后面（如写为 `data-parent=""`）。

名类	Type 类型	默认值	描述
parent	selector jQuery object DOM element	false	如果提供了一个选择器，然后当某个折叠块打开时，这个指定的父元素下面所有别的折叠块元素都将自动关闭（类似于传统的手风琴样式 - 这依赖 <code>card</code> 样式），属性必须在目标可折叠区域上定义。
toggle	boolean	true	在调用中折叠块元素。

方法

异步传输方法和转义

所有的 API 方法都是 **异步传输**和轮换，一旦 **事件发生（开始）**，在**结束之前**，它们会持续返回给调用者。另外在过渡组件上的方法将被忽略。

请参阅[我们的 API 文档](#)了解更多。

`.collapse(options)`

启用你的可折叠对象，通过 `object` 方法。

Copy


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

即发生 `shown.bs.collapse` 或 `hidden.bs.collapse` 事件前)返回给调用者。

```
.collapse('show')
```

显示可折叠元素，在可折叠元素实际显示之前 (即 `shown.bs.collapse` 事件发生之前)返回给调用者。

```
.collapse('hide')
```

隐藏可折叠元素，在可折叠元素实际上被隐藏之前 (即 `hidden.bs.collapse` 事件发生之前)返回给调用者。

```
.collapse('dispose')
```

销毁一个元素的折叠。

事件

Bootstrap 提供为折叠面板提供了一系列事件属性。

事件类别	描述
show.bs.collapse	当调用 <code>show</code> 方法时，会立即触发事件。
shown.bs.collapse	用户可见折叠面板中的块时，会触发此事件（需要等 CSS 加载过渡完成）。
hide.bs.collapse	当调用 <code>hide</code> 方法时，立即触发该事件。
hidden.bs.collapse	当折叠面板中的块对于用户完全隐藏时（需要等 CSS 加载过渡完成），会触发该事件）。

Copy

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

下拉菜单 (Dropdowns)

使用 Bootstrap 下拉插件用于显示切换你要展示的连接列表和更多内容的，或触发其它内容显示(覆盖)。

概览

弹出菜单是可触发的、上下文叠加显示链接列表和别的内容。它们可以与 Bootstrap 内置的弹出菜单 JavaScript 插件交互。通过点击触发，而不是通过鼠标悬停悬浮，这是[一种设计思维](#)。

下拉菜单控件依赖于第三方 [Popper.js](#) 插件实现，[Popper.js](#) 插件提供了动态定痊和 viewport 浏览器窗口大小监测，使用时请确保 [popper.min.js](#) 文件放在 Bootstrap JS 之前，或者使用 [bootstrap.bundle.min.js](#) / [bootstrap.bundle.js](#) 这两个文件，因为这两个文件中已经包含了 Popper.js。

如果你要自行编译 JS，记得 [包含 util.js](#)。

无障碍浏览提示(易用性)

[WAI ARIA](#) 标准定义了 `role="menu"` [插件](#)，但這是專門用於應用程式的功能表，它們觸發動作或功能。ARIA 菜单只能包含菜单列表、复选框、单选框、单选按钮组和子菜单。

Bootstrap 的下拉菜单则是设计为通用的，适用于各种情形和标记结构，如可以创建包含其它输入和表单控制项（如搜索栏位或登录表单）的下拉菜单，因此，Bootstrap 不希望（也不能自動添加）与 ARIA 菜单所需要的 `role` 和 `aria-` 属性，如果有必要请用户自行定义这类属性。

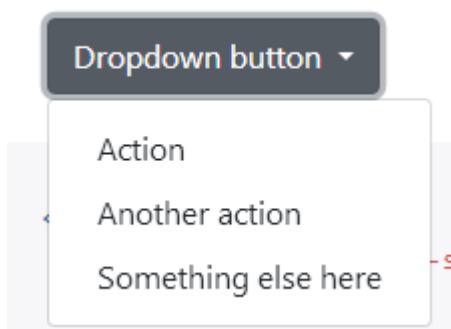
然而，Bootstrap 为大多数标准键盘功能操作加入了支持，如 `.dropdown-item` 支持光标移动选择单个子项、并使用关闭菜单的 快捷键功能。

示例

将下拉列表的切换（按钮或链接）和下拉菜单包含在 `.dropdown` 中，或者另外声明 `position: relative;` 元素; 可以从 `<a>` 或 `<button>` 触发下拉菜单，以适应你的使用的需求。

单一按钮的下拉菜单

任何一个 `.btn` 块都可以定义变更为下拉菜单，下面是两个使用 `<button>` 元素做下拉菜单的示例。

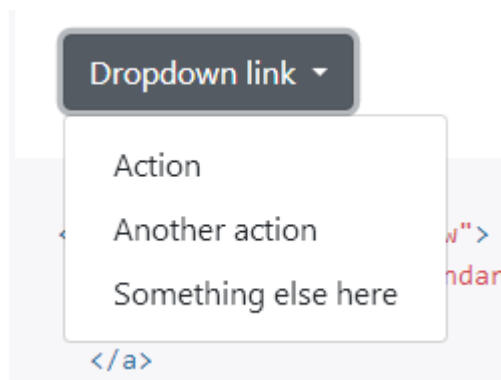


```

<div class="dropdown">
  <button class="btn btn-secondary dropdown-toggle" type="button"
id="dropdownMenuButton" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    Dropdown button
  </button>
  <div class="dropdown-menu" aria-labelledby="dropdownMenuButton">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <a class="dropdown-item" href="#">Something else here</a>
  </div>
</div>

```

使用 `<a>` 标签的下拉菜单：



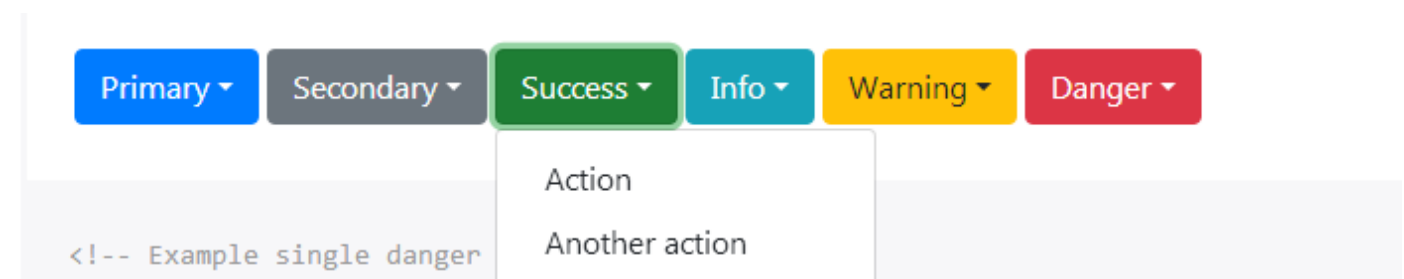
```

<div class="dropdown show">
  <a class="btn btn-secondary dropdown-toggle" href="#" role="button"
id="dropdownMenuLink" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    Dropdown link
  </a>

  <div class="dropdown-menu" aria-labelledby="dropdownMenuLink">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <a class="dropdown-item" href="#">Something else here</a>
  </div>
</div>

```

还可以自由引用 `.btn-prima` 等颜色及样式类来定义下拉菜单的外在表现：



```

<!-- Example single danger button -->
<div class="btn-group">
  <button type="button" class="btn btn-danger dropdown-toggle"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">

```

```

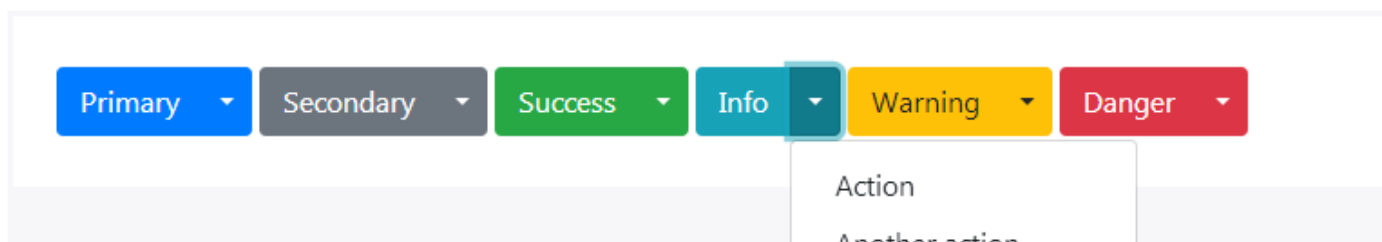
    Action
  </button>
  <div class="dropdown-menu">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <a class="dropdown-item" href="#">Something else here</a>
    <div class="dropdown-divider"></div>
    <a class="dropdown-item" href="#">Separated link</a>
  </div>
</div>

```

分裂式按钮下拉菜单

同样，可用与单个按钮下拉菜单近似的标记创建分裂式下拉菜单，注意添加了 `.dropdown-toggle-split` -插入此符号为下拉选项作适当的间隔（距）处理。

我们使用这个额外的 Class 样式，将插入符号两边水平 padding 减少了 25%，并移除了为默认下拉菜单添加的 `margin-left` 属性，这些额外的更改将插入符号集中在分裂式按钮中，并在主按钮旁边提供了适合的点击空间。



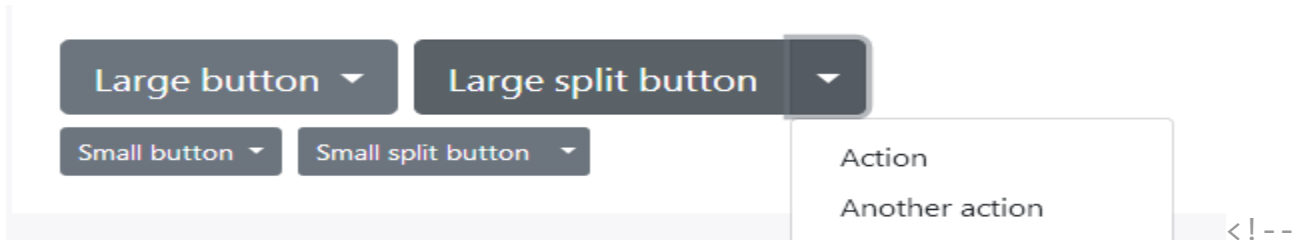
```

<!-- Example split danger button -->
<div class="btn-group">
  <button type="button" class="btn btn-danger">Action</button>
  <button type="button" class="btn btn-danger dropdown-toggle
dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    <span class="sr-only">Toggle Dropdown</span>
  </button>
  <div class="dropdown-menu">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <a class="dropdown-item" href="#">Something else here</a>
    <div class="dropdown-divider"></div>
    <a class="dropdown-item" href="#">Separated link</a>
  </div>
</div>

```

尺寸大小定义

下拉菜单有各种大小规格可以选用，包括预设及分裂式按钮下拉菜单。



Large button groups (default and split) -->

```
<div class="btn-group">
  <button class="btn btn-secondary btn-lg dropdown-toggle" type="button"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Large button
  </button>
  <div class="dropdown-menu">
    ...
  </div>
</div>
<div class="btn-group">
  <button class="btn btn-secondary btn-lg" type="button">
    Large button
  </button>
  <button type="button" class="btn btn-lg btn-secondary dropdown-toggle
dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    <span class="sr-only">Toggle Dropdown</span>
  </button>
  <div class="dropdown-menu">
    ...
  </div>
</div>
```

<!-- Small button groups (default and split) -->

```
<div class="btn-group">
  <button class="btn btn-secondary btn-sm dropdown-toggle" type="button"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Small button
  </button>
  <div class="dropdown-menu">
    ...
  </div>
</div>
<div class="btn-group">
  <button class="btn btn-secondary btn-sm" type="button">
    Small button
  </button>
  <button type="button" class="btn btn-sm btn-secondary dropdown-toggle
dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    <span class="sr-only">Toggle Dropdown</span>
  </button>
```

```

<div class="dropdown-menu">
  ...
</div>
</div>

```

变形-向上展开

增加 `.dropup` 样式，使下拉菜单向上展开。



```

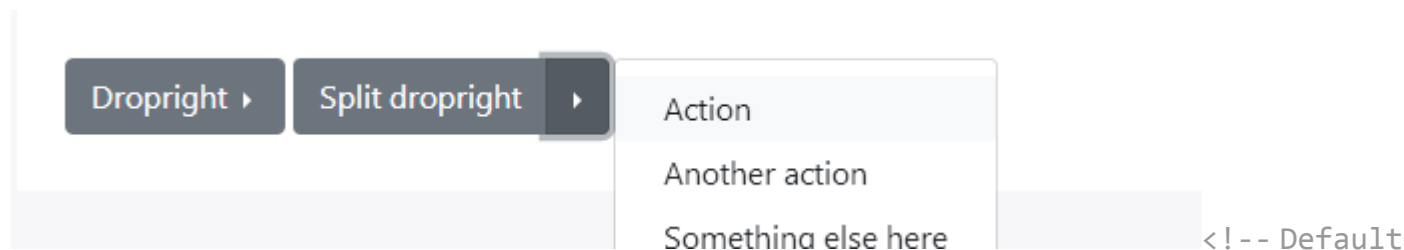
<!-- Default dropup button -->
<div class="btn-group dropup">
  <button type="button" class="btn btn-secondary dropdown-toggle"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Dropup
  </button>
  <div class="dropdown-menu">
    <!-- Dropdown menu links -->
  </div>
</div>

<!-- Split dropup button -->
<div class="btn-group dropup">
  <button type="button" class="btn btn-secondary">
    Split dropup
  </button>
  <button type="button" class="btn btn-secondary dropdown-toggle
dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    <span class="sr-only">Toggle Dropdown</span>
  </button>
  <div class="dropdown-menu">
    <!-- Dropdown menu links -->
  </div>
</div>

```

Dropright 右指向下拉菜单

通过将 `.dropright` 添加到父元素来触发元素左侧的下拉菜单。



```

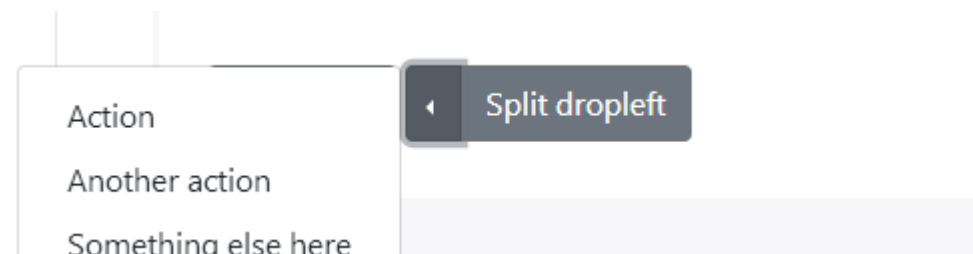
<!-- Default dropright button -->
<div class="btn-group dropright">
  <button type="button" class="btn btn-secondary dropdown-toggle"
    data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Dropright
  </button>
  <div class="dropdown-menu">
    <!-- Dropdown menu links -->
  </div>
</div>

<!-- Split dropright button -->
<div class="btn-group dropright">
  <button type="button" class="btn btn-secondary">
    Split dropright
  </button>
  <button type="button" class="btn btn-secondary dropdown-toggle
    dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
    aria-expanded="false">
    <span class="sr-only">Toggle Dropright</span>
  </button>
  <div class="dropdown-menu">
    <!-- Dropdown menu links -->
  </div>
</div>

```

Droprleft 左指向下拉菜单

通过将 `.dropleft` 添加到父元素来触发元素左侧的下拉菜单。



```

<!-- Default dropleft button -->
<div class="btn-group dropleft">
  <button type="button" class="btn btn-secondary dropdown-toggle"
    data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Dropleft
  </button>

```

```

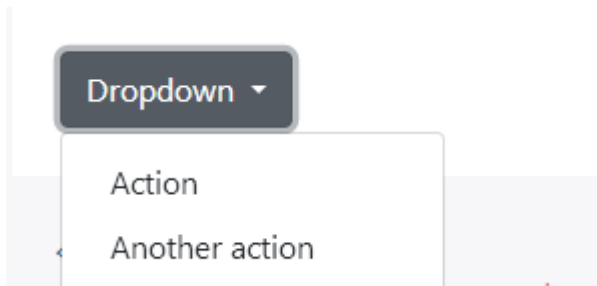
<div class="dropdown-menu">
  <!-- Dropdown menu links -->
</div>
</div>

<!-- Split dropleft button -->
<div class="btn-group">
  <div class="btn-group dropleft" role="group">
    <button type="button" class="btn btn-secondary dropdown-toggle dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
      <span class="sr-only">Toggle Dropleft</span>
    </button>
    <div class="dropdown-menu">
      <!-- Dropdown menu links -->
    </div>
  </div>
  <button type="button" class="btn btn-secondary">
    Split dropleft
  </button>
</div>

```

菜单项

旧版 Bootstrap(v3)下拉菜单中的子菜单项必须是链接，但 v4 不再是这种情况，现在你可选择使用 `<button>` 下拉列表中的元素，而不是仅仅 `<a>` 标签。



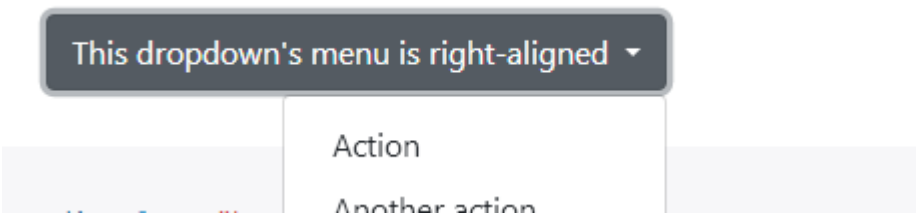
```

<div class="dropdown">
  <button class="btn btn-secondary dropdown-toggle" type="button" id="dropdownMenu2" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    Dropdown
  </button>
  <div class="dropdown-menu" aria-labelledby="dropdownMenu2">
    <button class="dropdown-item" type="button">Action</button>
    <button class="dropdown-item" type="button">Another action</button>
    <button class="dropdown-item" type="button">Something else here</button>
  </div>
</div>

```


菜单对齐

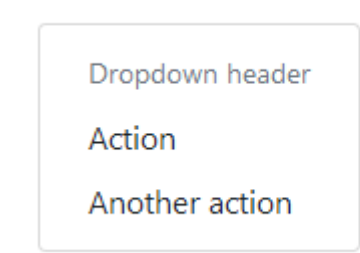
默认情况下，一个下拉菜单自动从顶部和左侧的父级 100% 定位。添加 `.dropdown-menu-right` 到 `.dropdown-menu` 右侧轻松对齐下拉菜单。



```
<div class="btn-group">
  <button type="button" class="btn btn-secondary dropdown-toggle"
    data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    This dropdown's menu is right-aligned
  </button>
  <div class="dropdown-menu dropdown-menu-right">
    <button class="dropdown-item" type="button">Action</button>
    <button class="dropdown-item" type="button">Another action</button>
    <button class="dropdown-item" type="button">Something else here</button>
  </div>
</div>
```

菜单标题

添加标题来标记任何下拉菜单中的操作部分。



```
<div class="dropdown-menu">
  <h6 class="dropdown-header">Dropdown header</h6>
  <a class="dropdown-item" href="#">Action</a>
  <a class="dropdown-item" href="#">Another action</a>
</div>
```

菜单分隔与分割线

使用分隔符分割相关菜单子项，呈现出分组和分割线效果。

Action

Another action

Something else here

Separated link

```
<div class="dropdown-menu">
  <a class="dropdown-item" href="#">Action</a>
  <a class="dropdown-item" href="#">Another action</a>
  <a class="dropdown-item" href="#">Something else here</a>
  <div class="dropdown-divider"></div>
  <a class="dropdown-item" href="#">Separated link</a>
</div>
```

菜单表单

将表单放在下拉菜单中，或将其放入下拉菜单中，并使用 [margin 或 padding](#) 通用 CSS 样式调整空间和位置。

Email address

email@example.com

Password

Password

☐ Remember me

Sign in

New around here? Sign up

Forgot password?

```
<div class="dropdown-menu">
  <form class="px-4 py-3">
    <div class="form-group">
      <label for="exampleDropdownFormEmail1">Email address</label>
      <input type="email" class="form-control" id="exampleDropdownFormEmail1"
placeholder="email@example.com">
    </div>
```

```

<div class="form-group">
  <label for="exampleDropdownFormPassword1">Password</label>
  <input type="password" class="form-control"
id="exampleDropdownFormPassword1" placeholder="Password">
</div>
<div class="form-check">
  <input type="checkbox" class="form-check-input" id="dropdownCheck" >
  <label class="form-check-label" for="dropdownCheck" >
    Remember me
  </label>
</div>
<button type="submit" class="btn btn-primary">Sign in</button>
</form>
<div class="dropdown-divider"></div>
<a class="dropdown-item" href="#">New around here? Sign up</a>
<a class="dropdown-item" href="#">Forgot password?</a>
</div>

```

```

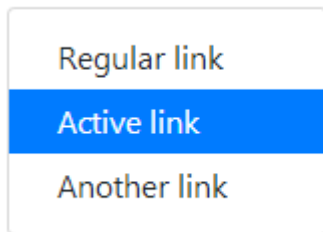
<form class="dropdown-menu p-4">
  <div class="form-group">
    <label for="exampleDropdownFormEmail2">Email address</label>
    <input type="email" class="form-control" id="exampleDropdownFormEmail2"
placeholder="email@example.com">
  </div>
  <div class="form-group">
    <label for="exampleDropdownFormPassword2">Password</label>
    <input type="password" class="form-control" id="exampleDropdownFormPassword2"
placeholder="Password">
  </div>
  <div class="form-check">
    <input type="checkbox" class="form-check-input" id="dropdownCheck2" >
    <label class="form-check-label" for="dropdownCheck2" >
      Remember me
    </label>
  </div>

```

```
<button type="submit" class="btn btn-primary">Sign in</button>
</form>
```

有效菜单项

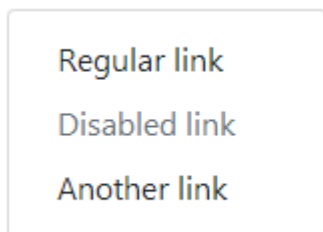
将 `.active` 添加到下拉菜单中的项目以将其设置为**活动状态**。



```
<div class="dropdown-menu">
  <a class="dropdown-item" href="#">Regular link</a>
  <a class="dropdown-item active" href="#">Active link</a>
  <a class="dropdown-item" href="#">Another link</a>
</div>
```

禁用菜单项

添加 `.disabled` 样式到下拉菜单中，**将下拉菜单子项用**（禁用后无法发生点击或选择事件，并不影响下拉菜单的其它动作，如下拉、其它子项样式）。



```
<div class="dropdown-menu">
  <a class="dropdown-item" href="#">Regular link</a>
  <a class="dropdown-item disabled" href="#">Disabled link</a>
  <a class="dropdown-item" href="#">Another link</a>
</div>
```

用例

通过数据属性或 JavaScript，下拉插件通过切换 `.show` 父列表项上的类来切换隐藏内容（下拉菜单）。该 `data-toggle="dropdown"` 属性依赖于在应用程序级别关闭下拉菜单，因此始终使用它是一个好主意。

在可触摸设备上，打开一个下拉菜单会将 `$.noop` `mouseover` 事件传递给 `<body>`，这是移动浏览器针对触屏的特殊处理方法（可见 [quirk in iOS' event delegation](#)），否则在下拉菜单之外的任何地方点击会无法触发关闭下拉菜单事件。一旦关闭下拉菜单，这些额外的事件会被立即移除。

通过事件属性

添加 `data-toggle="dropdown"` 在 A 链接或按钮上，以启用下拉菜单组件。

Copy

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

通过 JavaScript

通过 JavaScript 调用下菜单:

Copy

```
$('.dropdown-toggle').dropdown()
data-toggle="dropdown" 需要保留
```

无论你是采用 JavaScript 或者 data-api 调用下拉菜单，`data-toggle="dropdown"` 总要保留在下拉列表的触发器元素中。

选项

可以通过数据属性或 JavaScript 传递选项。对于数据属性，选项名称追加到 `data-`，如：`data-offset=""`。

名称	Type 类型	Default 默认值	描述说明
offset	number string function	0	下拉菜单相对于目标的偏移，参考 Popper.js 偏移 文档。
flip	boolean	true	允许下拉菜单重叠到其相关元素上，参考 Popper.js 触发 文档。

方法

Method	Description
<code>\$('.dropdown-toggle').dropdown()</code>	给导航栏或分页启用下拉菜单功能。
<code>\$('.dropdown-toggle').dropdown('update')</code>	更新下拉列表的位置。
<code>\$('.dropdown-toggle').dropdown('dispose')</code>	销毁一个元素的下拉菜单。

事件

All dropdown events are fired at the `.dropdown-menu`'s parent element and have a `relatedTarget` property, whose value is the toggling anchor element.

Event 事件	描述说明
<code>show.bs.dropdown</code>	当调用 <code>show</code> 显示方法时，此事件会立即触发。
<code>shown.bs.dropdown</code>	当下拉菜单对用户可见时，会触发此事件（将等待 CSS 转换完成）。
<code>hide.bs.dropdown</code>	当调用隐藏实例方法时，会立即触发此事件。
<code>hidden.bs.dropdown</code>	当下拉菜单从用户隐藏完毕时，会触发此事件（将等待 CSS 转换完成）。

Copy

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

表单 (Forms)

Bootstrap 提供了一些表单控件样式、布局选项，以及用来创建广泛多样化的的表单的自定义组件，以下是示例和使用指南。

概览

Bootstrap 的表单控件使用了 [CSS 样式重置](#)，使用这些 Class 类来自定义显示，以便跨越浏览器和设备获得一致的呈现。

确保在输入框上使用正确的 `type` 属性(如 `email` 用于电子邮件地址或 `number` 用于数字录入) 从而利用较新的录入控制，包括诸如电子邮件验证、号码选择等。

以下是演示 Bootstrap 表单样式的一个经典示例展示，并推荐继续向下阅读有关 class 类、表单布局等文档。

Email address

We'll never share your email with anyone else.

Password

☐ Check me out

```
<form>
  <div class="form-group">
    <label for="exampleInputEmail1">Email address</label>
    <input type="email" class="form-control" id="exampleInputEmail1"
aria-describedby="emailHelp" placeholder="Enter email">
    <small id="emailHelp" class="form-text text-muted">We'll never share your email
with anyone else.</small>
  </div>
  <div class="form-group">
    <label for="exampleInputPassword1">Password</label>
    <input type="password" class="form-control" id="exampleInputPassword1"
placeholder="Password">
  </div>
  <div class="form-check">
    <input type="checkbox" class="form-check-input" id="exampleCheck1">
    <label class="form-check-label" for="exampleCheck1">Check me out</label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

表单控件

文本控件（如 `<input>`、`<select>`、`<textarea>`）统一采用 `.form-control` 样式进行处理优化，包括常规外观、focus 选（点）中状态、尺寸大小等。

推荐浏览 [自定义表单](#) 以进一步了解 `<select>` 控件的风格设计。

Email address

name@example.com

Example select

1

Example multiple select

1
2
3
4

Example textarea

```
<form>
  <div class="form-group">
    <label for="exampleFormControlInput1">Email address</label>
    <input type="email" class="form-control" id="exampleFormControlInput1"
placeholder="name@example.com">
  </div>
  <div class="form-group">
    <label for="exampleFormControlSelect1">Example select</label>
    <select class="form-control" id="exampleFormControlSelect1">
      <option>1</option>
      <option>2</option>
      <option>3</option>
      <option>4</option>
      <option>5</option>
    </select>
  </div>
  <div class="form-group">
    <label for="exampleFormControlSelect2">Example multiple select</label>
    <select multiple class="form-control" id="exampleFormControlSelect2">
      <option>1</option>
      <option>2</option>
      <option>3</option>
      <option>4</option>
      <option>5</option>
    </select>
  </div>
  <div class="form-group">
    <label for="exampleFormControlTextarea1">Example textarea</label>
```



```

    <textarea class="form-control" id="exampleFormControlTextarea1"
rows="3"></textarea>
  </div>
</form>

```

对于 input 文件选择控件，Bootstrap v4 采用 `.form-control-file` 取代了 `.form-control`。

Example file input

选择文件

 未选择任何文件

```

<form>
  <div class="form-group">
    <label for="exampleFormControlFile1">Example file input</label>
    <input type="file" class="form-control-file" id="exampleFormControlFile1">
  </div>
</form>

```

大小规格

使用 `.form-control-lg` 和 `.form-control-sm` 属性定控件大小高度。

.form-control-lg

Default input

.form-control-sm

```

<input class="form-control form-control-lg" type="text"
placeholder=".form-control-lg">
<input class="form-control" type="text" placeholder="Default input">
<input class="form-control form-control-sm" type="text"
placeholder=".form-control-sm">

```

Large select

Default select

Small select

```

<select class="form-control form-control-lg">
  <option>Large select</option>
</select>
<select class="form-control">
  <option>Default select</option>
</select>

```

```
<select class="form-control form-control-sm">
  <option>Small select</option>
</select>
```

只读属性

在 `input` 控件上增加 `readonly`（布尔值）标签定义，以防止修改 `input` 中的值。仅能阅读的 `input` 控件显示较淡（就象禁用的输入框），但保留鼠标效果。

Readonly input here...

```
<input class="form-control" type="text" placeholder="Readonly input here..."
readonly>
```

只读纯文本

如果你希望将 `<input readonly>` 属性进一步处理，显示为纯文本（没有控件框），你只要引用 `.form-control-plaintext` class 样式，就能移除预设的表单样式，并保留适当的边距和填充间隙。

Email email@example.com

Password Password

```
<form>
  <div class="form-group row">
    <label for="staticEmail" class="col-sm-2 col-form-label">Email</label>
    <div class="col-sm-10">
      <input type="text" readonly class="form-control-plaintext" id="staticEmail"
value="email@example.com">
    </div>
  </div>
  <div class="form-group row">
    <label for="inputPassword" class="col-sm-2 col-form-label">Password</label>
    <div class="col-sm-10">
      <input type="password" class="form-control" id="inputPassword"
placeholder="Password">
    </div>
  </div>
</form>
```

email@example.com

Password



Confirm identity

```

<form class="form-inline mb-2">
  <div class="form-group">
    <label for="staticEmail2" class="sr-only">Email</label>
    <input type="text" readonly class="form-control-plaintext" id="staticEmail2"
value="email@example.com">
  </div>
  <div class="form-group mx-sm-3 mb-2">
    <label for="inputPassword2" class="sr-only">Password</label>
    <input type="password" class="form-control" id="inputPassword2"
placeholder="Password">
  </div>
  <button type="submit" class="btn btn-primary mb-2">Confirm identity</button>
</form>

```

复选框与单选框

使用 `.form-check` 可以格式化复选框和单选框按钮，用以改进它们的默认布局 and 动作呈现，复选框用于在列表中选择一个或多个选项，单选框则用于许许多多选项中选择一个。

复选框和单选框也是可以禁用的，只要 `not-allowed` 在父级的悬停上提供定义，`<label>` 需要将该 `.disabled` 类添加到父级 `.form-check`，同时控件也会淡化文字颜色以灰色显示禁用状态。

默认堆叠

默认情况下，同级任意数量的复选框与单选框按钮垂直堆叠，并与 `.form-check` 有间隙隔离。

- ☐ Option one is this and that—be sure to include why it's great
☐ Option two is disabled

```

<div class="form-check">
  <input class="form-check-input" type="checkbox" value="" id="defaultCheck1">
  <label class="form-check-label" for="defaultCheck1">
    Option one is this and that—be sure to include why it's great
  </label>
</div>
<div class="form-check">
  <input class="form-check-input" type="checkbox" value="" id="defaultCheck2"
disabled>
  <label class="form-check-label" for="defaultCheck2">
    Option two is disabled
  </label>
</div>

```

- Option one is this and that—be sure to include why it's great
- Option two can be something else and selecting it will deselect option one
- Option three is disabled

```
<div class="form-check">
  <input class="form-check-input" type="radio" name="exampleRadios"
id="exampleRadios1" value="option1" checked>
  <label class="form-check-label" for="exampleRadios1">
    Option one is this and that—be sure to include why it's great
  </label>
</div>
<div class="form-check">
  <input class="form-check-input" type="radio" name="exampleRadios"
id="exampleRadios2" value="option2">
  <label class="form-check-label" for="exampleRadios2">
    Option two can be something else and selecting it will deselect option one
  </label>
</div>
<div class="form-check disabled">
  <input class="form-check-input" type="radio" name="exampleRadios"
id="exampleRadios3" value="option3" disabled>
  <label class="form-check-label" for="exampleRadios3">
    Option three is disabled
  </label>
</div>
```

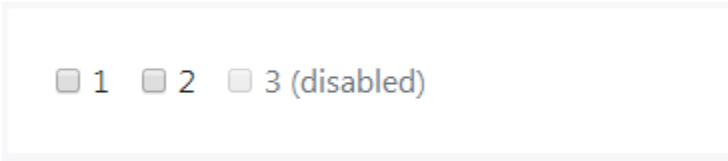
水平排列

通过添加 `.form-check-inline` 到任何一个组，会使 加到任何 `.form-check` 中的选取框平行排列。

☐ 1 ☐ 2 ☐ 3 (disabled)

```
<div class="form-check form-check-inline">
  <input class="form-check-input" type="checkbox" id="inlineCheckbox1"
value="option1">
  <label class="form-check-label" for="inlineCheckbox1">1</label>
</div>
<div class="form-check form-check-inline">
  <input class="form-check-input" type="checkbox" id="inlineCheckbox2"
value="option2">
  <label class="form-check-label" for="inlineCheckbox2">2</label>
</div>
<div class="form-check form-check-inline">
```

```
<input class="form-check-input" type="checkbox" id="inlineCheckbox3"
value="option3" disabled>
<label class="form-check-label" for="inlineCheckbox3">3 (disabled)</label>
</div>
```



```
<div class="form-check form-check-inline">
  <input class="form-check-input" type="radio" name="inlineRadioOptions"
id="inlineRadio1" value="option1">
  <label class="form-check-label" for="inlineRadio1">1</label>
</div>
<div class="form-check form-check-inline">
  <input class="form-check-input" type="radio" name="inlineRadioOptions"
id="inlineRadio2" value="option2">
  <label class="form-check-label" for="inlineRadio2">2</label>
</div>
<div class="form-check form-check-inline">
  <input class="form-check-input" type="radio" name="inlineRadioOptions"
id="inlineRadio3" value="option3" disabled>
  <label class="form-check-label" for="inlineRadio3">3 (disabled)</label>
</div>
```

没有标签

添加 `.position-static` 到 `.form-check` 选择器上，可以实现没有文本的输入，记住：仍然要为辅助浏览（友好访问）提供相应标签，如使用 `aria-label` 定义。

```
<div class="form-check">
  <input class="form-check-input position-static" type="checkbox"
id="blankCheckbox" value="option1" aria-label="...">
</div>
<div class="form-check">
  <input class="form-check-input position-static" type="radio" name="blankRadio"
id="blankRadio1" value="option1" aria-label="...">
</div>
```

布局

自从 Bootstrap 使用 `display: block` 和 `width: 100%` 在全部的 `input` 控件上后，表单默认都是基于垂直堆叠排列的，可以使用其它 Class 类来改变表单的布局。

表单组

`.form-group` 群组可以向为表单赋予一些结构样式，其唯一目的是提供标签的控制配对以及 `margin-bottom` 属性，由于它是一个 class 选择器，你可以在 `<fieldset>`、`<div>` 或任何其它元素中使用它。

Example label

Example input

Another label

Another input

```
<form>
  <div class="form-group">
    <label for="formGroupExampleInput">Example label</label>
    <input type="text" class="form-control" id="formGroupExampleInput"
placeholder="Example input">
  </div>
  <div class="form-group">
    <label for="formGroupExampleInput2">Another label</label>
    <input type="text" class="form-control" id="formGroupExampleInput2"
placeholder="Another input">
  </div>
</form>
```

表单栅格排列

可使用我们的栅格系统构建更复杂的表单，包括建立多列、多种宽度和其它对齐选项的布局。

First name

Last name

```
<form>
  <div class="row">
    <div class="col">
      <input type="text" class="form-control" placeholder="First name">
    </div>
    <div class="col">
      <input type="text" class="form-control" placeholder="Last name">
    </div>
  </div>
</form>
```

表格式排列

你也可以使用 `.form-row` 来取代 `.row`（它们二者很多时候可以互换使用），因为 `.form-row` 提供更小的沟槽缝隙。

First name

Last name

```
<form>
  <div class="form-row">
    <div class="col">
```

```

    <input type="text" class="form-control" placeholder="First name">
  </div>
  <div class="col">
    <input type="text" class="form-control" placeholder="Last name">
  </div>
</div>
</form>

```

还可以使用栅格系统建立更复杂的布局。

Email

Password

Address

Address 2

City

State

Zip

☐ Check me out

```

<form>
  <div class="form-row">
    <div class="form-group col-md-6">
      <label for="inputEmail4">Email</label>
      <input type="email" class="form-control" id="inputEmail4"
placeholder="Email">
    </div>
    <div class="form-group col-md-6">
      <label for="inputPassword4">Password</label>
      <input type="password" class="form-control" id="inputPassword4"
placeholder="Password">
    </div>
  </div>
  <div class="form-group">
    <label for="inputAddress">Address</label>
    <input type="text" class="form-control" id="inputAddress" placeholder="1234
Main St">
  </div>
  <div class="form-group">
    <label for="inputAddress2">Address 2</label>
    <input type="text" class="form-control" id="inputAddress2"
placeholder="Apartment, studio, or floor">
  </div>
  <div class="form-row">

```

```

<div class="form-group col-md-6">
  <label for="inputCity">City</label>
  <input type="text" class="form-control" id="inputCity">
</div>
<div class="form-group col-md-4">
  <label for="inputState">State</label>
  <select id="inputState" class="form-control">
    <option selected>Choose...</option>
    <option>...</option>
  </select>
</div>
<div class="form-group col-md-2">
  <label for="inputZip">Zip</label>
  <input type="text" class="form-control" id="inputZip">
</div>
</div>
<div class="form-group">
  <div class="form-check">
    <input class="form-check-input" type="checkbox" id="gridCheck">
    <label class="form-check-label" for="gridCheck">
      Check me out
    </label>
  </div>
</div>
<button type="submit" class="btn btn-primary">Sign in</button>
</form>

```

垂直排列表单

通过添加 `.row` class 类，并使用 `.col-*-*` 等栅格组件来指定标签和宽度，可以建立起水平表单。确保添加 `.col-form-label` 到您 `<label>` 上，以便他们垂直居中与他们相关的表单控件。`<legend>` 元素，可以 `.col-form-legend` 样式定义，与普通 `<label>` 元素相似。

Email	<input type="text" value="Email"/>
Password	<input type="password" value="Password"/>
Radios	<input checked="" type="radio"/> First radio <input type="radio"/> Second radio <input type="disabled" value="disabled"/> Third disabled radio
Checkbox	<input type="checkbox"/> Example checkbox
<input type="button" value="Sign in"/>	

```

<form>
  <div class="form-group row">
    <label for="inputEmail3" class="col-sm-2 col-form-label">Email</label>
    <div class="col-sm-10">

```



```

        <input type="email" class="form-control" id="inputEmail3"
placeholder="Email">
    </div>
</div>
<div class="form-group row">
    <label for="inputPassword3" class="col-sm-2 col-form-label">Password</label>
    <div class="col-sm-10">
        <input type="password" class="form-control" id="inputPassword3"
placeholder="Password">
    </div>
</div>
<fieldset class="form-group">
    <div class="row">
        <legend class="col-form-label col-sm-2 pt-0">Radios</legend>
        <div class="col-sm-10">
            <div class="form-check">
                <input class="form-check-input" type="radio" name="gridRadios"
id="gridRadios1" value="option1" checked>
                <label class="form-check-label" for="gridRadios1">
                    First radio
                </label>
            </div>
            <div class="form-check">
                <input class="form-check-input" type="radio" name="gridRadios"
id="gridRadios2" value="option2">
                <label class="form-check-label" for="gridRadios2">
                    Second radio
                </label>
            </div>
            <div class="form-check disabled">
                <input class="form-check-input" type="radio" name="gridRadios"
id="gridRadios3" value="option3" disabled>
                <label class="form-check-label" for="gridRadios3">
                    Third disabled radio
                </label>
            </div>
        </div>
    </div>
</fieldset>
<div class="form-group row">
    <div class="col-sm-2">Checkbox</div>
    <div class="col-sm-10">
        <div class="form-check">
            <input class="form-check-input" type="checkbox" id="gridCheck1">
            <label class="form-check-label" for="gridCheck1">
                Example checkbox
            </label>
        </div>
    </div>
</div>

```

```

</div>
<div class="form-group row">
  <div class="col-sm-10">
    <button type="submit" class="btn btn-primary">Sign in</button>
  </div>
</div>
</form>

```

垂直排列表单尺寸规格定义

使用.col-form-label-sm、.col-form-label-lg 到 <label>上，可以定义控件大小，还有 .form-control-lg、.form-control-sm 样式也起相应作用。

Email	col-form-label-sm
Email	col-form-label
Email	col-form-label-lg

```

<form>
  <div class="form-group row">
    <label for="colFormLabelSm" class="col-sm-2 col-form-label col-form-label-sm">Email</label>
    <div class="col-sm-10">
      <input type="email" class="form-control form-control-sm" id="colFormLabelSm" placeholder="col-form-label-sm">
    </div>
  </div>
  <div class="form-group row">
    <label for="colFormLabel" class="col-sm-2 col-form-label">Email</label>
    <div class="col-sm-10">
      <input type="email" class="form-control" id="colFormLabel" placeholder="col-form-label">
    </div>
  </div>
  <div class="form-group row">
    <label for="colFormLabelLg" class="col-sm-2 col-form-label col-form-label-lg">Email</label>
    <div class="col-sm-10">
      <input type="email" class="form-control form-control-lg" id="colFormLabelLg" placeholder="col-form-label-lg">
    </div>
  </div>
</form>

```

栅格式列尺寸定义

如前面的示例所示，我們的栅格系統允許您将任意数量的 `col` 放置在 `form-row` 中，它们会在自动进行宽度分割（根据您的栅格定义），从而定义某列更窄或某列更宽（其余列享受流式布局宽度定义）。

```
<form>
  <div class="form-row">
    <div class="col-7">
      <input type="text" class="form-control" placeholder="City">
    </div>
    <div class="col">
      <input type="text" class="form-control" placeholder="State">
    </div>
    <div class="col">
      <input type="text" class="form-control" placeholder="Zip">
    </div>
  </div>
</form>
```

自动调整大小

下面的示例使用一个 flexbox 弹性布局垂直居中的内容，我们将 `col` 改为 `col-auto`，这样的列只占用本身内容所需要的宽度，换句话说列的大小就是内容的大小

☐ Remember me

```
<form>

  <div class="form-row align-items-center">
    <div class="col-auto">
      <label class="sr-only" for="inlineFormInput">Name</label>
      <input type="text" class="form-control mb-2" id="inlineFormInput"
placeholder="Jane Doe">
    </div>
    <div class="col-auto">
      <label class="sr-only" for="inlineFormInputGroup">Username</label>
      <div class="input-group mb-2">
        <div class="input-group-prepend">
          <div class="input-group-text">@</div>
        </div>
        <input type="text" class="form-control" id="inlineFormInputGroup"
placeholder="Username">
      </div>
    </div>
  </div>
```

```

<div class="col-auto">
  <div class="form-check mb-2">
    <input class="form-check-input" type="checkbox" id="autoSizingCheck">
    <label class="form-check-label" for="autoSizingCheck">
      Remember me
    </label>
  </div>
</div>
<div class="col-auto">
  <button type="submit" class="btn btn-primary mb-2">Submit</button>
</div>
</div>
</form>

```

然后，你可以重新混合不同大小的 class 样式。

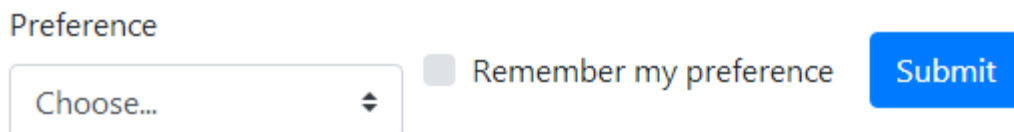
☐ Remember me

```

<form>
  <div class="form-row align-items-center">
    <div class="col-sm-3 my-1">
      <label class="sr-only" for="inlineFormInputName">Name</label>
      <input type="text" class="form-control" id="inlineFormInputName"
placeholder="Jane Doe">
    </div>
    <div class="col-sm-3 my-1">
      <label class="sr-only" for="inlineFormInputGroupUsername">Username</label>
      <div class="input-group">
        <div class="input-group-prepend">
          <div class="input-group-text">@</div>
        </div>
        <input type="text" class="form-control" id="inlineFormInputGroupUsername"
placeholder="Username">
      </div>
    </div>
    <div class="col-auto my-1">
      <div class="form-check">
        <input class="form-check-input" type="checkbox" id="autoSizingCheck2">
        <label class="form-check-label" for="autoSizingCheck2">
          Remember me
        </label>
      </div>
    </div>
    <div class="col-auto my-1">
      <button type="submit" class="btn btn-primary">Submit</button>
    </div>
  </div>
</form>

```

也支持 [自定义表单](#) 控件。




```
<form>
  <div class="form-row align-items-center">
    <div class="col-auto my-1">
      <label class="mr-sm-2" for="inlineFormCustomSelect">Preference</label>
      <select class="custom-select mr-sm-2" id="inlineFormCustomSelect">
        <option selected>Choose...</option>
        <option value="1">One</option>
        <option value="2">Two</option>
        <option value="3">Three</option>
      </select>
    </div>
    <div class="col-auto my-1">
      <div class="custom-control custom-checkbox mr-sm-2">
        <input type="checkbox" class="custom-control-input"
id="customControlAutosizing">
        <label class="custom-control-label"
for="customControlAutosizing">Remember my preference</label>
      </div>
    </div>
    <div class="col-auto my-1">
      <button type="submit" class="btn btn-primary">Submit</button>
    </div>
  </div>
</form>
```

内联式表单

使用 `.form-inline` 样式在单个水平行上显示一系列标签，表单控件和按钮。内联表单中的表单控件与默认状态略有不同：

- 基于 `display: flex` 控件组件，并允许您使用 [间隙隔离](#) 和 [flexbox](#) 弹性布局样式。
- 控制组件和 `input` 接受 `width: auto` 以覆盖预设的 `width: 100%`。
- 控制组件只会在 **viewport 576px 宽度** 时才会显示在行内，以便在移动设备上完整呈现。

你需要手机添加宽度或对齐在单个的控制元件上，通过 [间隙隔离定义](#) (如下所示)，最后一定要将 `<label>` 包含在每个表单控制元件内，即使你需要使用 `.sr-only` 从非正常屏幕访问它或隐藏它。



```
<form class="form-inline">
  <label class="sr-only" for="inlineFormInputName2">Name</label>
```

```

<input type="text" class="form-control mb-2 mr-sm-2" id="inlineFormInputName2"
placeholder="Jane Doe">

<label class="sr-only" for="inlineFormInputGroupUsername2">Username</label>
<div class="input-group mb-2 mr-sm-2">
  <div class="input-group-prepend">
    <div class="input-group-text">@</div>
  </div>
  <input type="text" class="form-control" id="inlineFormInputGroupUsername2"
placeholder="Username">
</div>

<div class="form-check mb-2 mr-sm-2">
  <input class="form-check-input" type="checkbox" id="inlineFormCheck">
  <label class="form-check-label" for="inlineFormCheck">
    Remember me
  </label>
</div>

<button type="submit" class="btn btn-primary mb-2">Submit</button>
</form>

```

自定义表单和 **select** 选择控件也是支持的。

```

<form class="form-inline">
  <label class="my-1 mr-2" for="inlineFormCustomSelectPref">Preference</label>
  <select class="custom-select my-1 mr-sm-2" id="inlineFormCustomSelectPref">
    <option selected>Choose...</option>
    <option value="1">One</option>
    <option value="2">Two</option>
    <option value="3">Three</option>
  </select>

  <div class="custom-control custom-checkbox my-1 mr-sm-2">
    <input type="checkbox" class="custom-control-input" id="customControlInline">
    <label class="custom-control-label" for="customControlInline">Remember my
preference</label>
  </div>

  <button type="submit" class="btn btn-primary my-1">Submit</button>
</form>

```

隐藏的标签替代

如果您没有为每个 `input` 添加标签，屏幕阅读器等辅助技术将对您的表单有困难。对于这些内联表单，您可以使用 `.sr-only` 类隐藏标签。同时我们还提供了辅助技术标签以外的解决方法，例如：`aria-label`、`aria-labelledby` 或 `title` 属性，如果这些都不存在，辅助技术可以使用 `placeholder` 的属性（如果它存在），但注意：不推荐使用 `placeholder` 替代其它标签方法。

表单下方帮助提示文本

可以使用 `.form-text`（以前称为 `.help-block` 在 v3 中）创建表单中的块级帮助文本。可以使用任何内联 HTML 元素和通用样式（如 `.text-muted`）灵活的展示帮助提示文本。

帮助提示文本需要与表单控件关联

帮助提示文本应该与使用该 `aria-describedby` 与表单控制元素相关联，这将确保辅助技术（如读屏器）在用户 focus 点中或 `input` 输入时了解此帮助提示文字。

`input` 下方的帮助文字可以用 `.form-text`，包括 `display: block` 并添加一些 `margin-top` 以便与上面 `input` 框有所间隔。

Password

Your password must be 8-20 characters long, contain letters and numbers, and must not contain spaces, special characters, or emoji.

```
<label for="inputPassword5">Password</label>
```

```
<input type="password" id="inputPassword5" class="form-control"
aria-describedby="passwordHelpBlock">
<small id="passwordHelpBlock" class="form-text text-muted">
```

Your password must be 8-20 characters long, contain letters and numbers, and must not contain spaces, special characters, or emoji.

```
</small>
```

行内也可以嵌入任意典型的 HTML 元素（无是 `<small>`、``，或其它样式）。

Password

Must be 8-20 characters long.

```
<form class="form-inline">
  <div class="form-group">
    <label for="inputPassword6">Password</label>
    <input type="password" id="inputPassword6" class="form-control mx-sm-3"
aria-describedby="passwordHelpInline">
    <small id="passwordHelpInline" class="text-muted">
      Must be 8-20 characters long.
    </small>
  </div>
</form>
```

禁用表单

添加 `disabled` 布尔值属性到 `input` 上，就能防止使用者操作并看起来更灰淡。

Copy

```
<input class="form-control" id="disabledInput" type="text" placeholder="Disabled input here..." disabled>
```

Add the `disabled` attribute to a `<fieldset>` to disable all the controls within.

Disabled input

Disabled input

Disabled select menu

Disabled select

☐ Can't check this

Submit

```
<form>
  <fieldset disabled>
    <div class="form-group">
      <label for="disabledTextInput">Disabled input</label>
      <input type="text" id="disabledTextInput" class="form-control"
placeholder="Disabled input">
    </div>
    <div class="form-group">
      <label for="disabledSelect">Disabled select menu</label>
      <select id="disabledSelect" class="form-control">
        <option>Disabled select</option>
      </select>
    </div>
    <div class="form-check">
      <input class="form-check-input" type="checkbox" id="disabledFieldsetCheck"
disabled>
      <label class="form-check-label" for="disabledFieldsetCheck">
        Can't check this
      </label>
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
  </fieldset>
</form>
```

附加说明与锚

默认情况下，浏览器会将`<fieldset disabled>`中所有的表单控制项（`<input>`、`<select>`、`<button>` 元素）视为禁用，以防止键盘鼠标与它们发生交互，但是如果你的表单还包含了 `<a ... class="btn btn-*">` 元素，则只会给出一种 `pointer-events: none` 的样式，关于按钮禁用（特别是 [btn 按钮禁用状态](#)（特别是在提交元素的子部份），此 CSS 属性尚未标准化（被各浏览器厂商广泛支持），Opera 18 及更低版本或 IE10 中完全不兼容，并且不会阻止键盘使用者 focus 或启动这些链接。为了安全起见，请自行使用自定义 JavaScript 来禁用这些链接。

跨浏览器兼容性支持

虽然 Bootstrap 会在所有的浏览器应用这些样式，但是 Internet Explorer 11 及更低版本 IE 不完全支持 `<fieldset>` 标签上的 `disabled` 属性，故在 IE 中请使用自定义 JavaScript 脚本来禁用标签（已经不存在 IE12，未来的 IE 已更名 Edge 浏览器-译者注）。

验证

通过 HTML5 表单验证可以在我们[支持的浏览器](#)中为你的用户提供可操作的反馈。从浏览器中选择默认的验证反馈，或者使用我们内置的 Class 样式和启动 JavaScript 实现自定义消息。

我们**强烈推荐使用自定义方式**-因为浏览器默认不会向屏幕阅读者提供友好反馈。

运行原理

Bootstrap 中验证机制如下：

- HTML 表单验证通过 CSS 的两个伪类应用 `:invalid` 以及 `:valid`。它适用于 `<input>`、`<select>`、`<textarea>` 元素。
- Bootstrap `:invalid` 和 `:valid` 样式在 `.was-validated`，通常会反馈到 `<form>`，否则必填的内容在加载的过程中会显示无效，这个方法你呆以选择在任何时候启用它（通过是在尝试提交表单后）。
- 作为备用，`.is-invalid` 和 `.is-valid` 类别可以用来代替伪类用于服务器端验证。他们不需要[server side validation](#)定义在父层上。
- 由于 CSS 的工作原理限制，我们无法（现在）将样式应用于 DOM 中的表单控制组件之前的 `<label>` 上，而不需要使用自定义 JavaScript。
- 所有现代浏览器都支持[约束验证 API](#)，这是一系列用于验证表单控件的 JavaScript 方法。
- 反馈消息可能会使用[浏览器默认值](#)（每个浏览器不同，通过私有 CSS 样式）或其他 HTML 和 CSS 的自定义反馈样式。
- 你可以在 JavaScript 中 使用 `setCustomValidity` 提供有效自定义消息。

请参与以下的自定义表单验证样式、服务器端验证 Class 和浏览器默认值。

自定义样式

对于自定义 Bootstrap 表单验证消息，您需要将 `novalidate` 属性添加到您的 `<form>`。这将禁用浏览器默认的反馈工具提示，但仍提供对 JavaScript 中的表单验证 API 有效支持。尝试提交以下表格；我们的 JavaScript 将拦截提交按钮并向您传递反馈：

尝试提交时，您将看到 `:invalid` 和 `:valid` 的样式应用于表单控件。

When attempting to submit, you'll see the styles applied to your form controls.

First name	Last name	Username
<input type="text" value="Mark"/>	<input type="text" value="Otto"/>	<input type="text" value="@ Username"/>

City	State	Zip
<input type="text" value="City"/>	<input type="text" value="State"/>	<input type="text" value="Zip"/>

☐ Agree to terms and conditions

Submit form

```
<form class="needs-validation" novalidate>
  <div class="form-row">
    <div class="col-md-4 mb-3">
      <label for="validationCustom01">First name</label>
      <input type="text" class="form-control" id="validationCustom01"
placeholder="First name" value="Mark" required>
      <div class="valid-feedback">
        Looks good!
      </div>
    </div>
    <div class="col-md-4 mb-3">
      <label for="validationCustom02">Last name</label>
      <input type="text" class="form-control" id="validationCustom02"
placeholder="Last name" value="Otto" required>
      <div class="valid-feedback">
        Looks good!
      </div>
    </div>
    <div class="col-md-4 mb-3">
      <label for="validationCustomUsername">Username</label>
      <div class="input-group">
        <div class="input-group-prepend">
          <span class="input-group-text" id="inputGroupPrepend">@</span>
        </div>
        <input type="text" class="form-control" id="validationCustomUsername"
placeholder="Username" aria-describedby="inputGroupPrepend" required>
        <div class="invalid-feedback">
          Please choose a username.
        </div>
      </div>
    </div>
  </div>
  <div class="form-row">
    <div class="col-md-6 mb-3">
      <label for="validationCustom03">City</label>
      <input type="text" class="form-control" id="validationCustom03"
placeholder="City" required>
      <div class="invalid-feedback">
        Please provide a valid city.
      </div>
    </div>
  </div>
</form>
```

```

    </div>
</div>
<div class="col-md-3 mb-3">
  <label for="validationCustom04">State</label>
  <input type="text" class="form-control" id="validationCustom04"
placeholder="State" required>
  <div class="invalid-feedback">
    Please provide a valid state.
  </div>
</div>
<div class="col-md-3 mb-3">
  <label for="validationCustom05">Zip</label>
  <input type="text" class="form-control" id="validationCustom05"
placeholder="Zip" required>
  <div class="invalid-feedback">
    Please provide a valid zip.
  </div>
</div>
</div>
<div class="form-group">
  <div class="form-check">
    <input class="form-check-input" type="checkbox" value="" id="invalidCheck"
required>
    <label class="form-check-label" for="invalidCheck">
      Agree to terms and conditions
    </label>
    <div class="invalid-feedback">
      You must agree before submitting.
    </div>
  </div>
</div>
<button class="btn btn-primary" type="submit">Submit form</button>
</form>

```

```

<script>
// Example starter JavaScript for disabling form submissions if there are invalid
fields
(function() {
  'use strict';
  window.addEventListener('load', function() {
    // Fetch all the forms we want to apply custom Bootstrap validation styles to
    var forms = document.getElementsByClassName('needs-validation');
    // Loop over them and prevent submission
    var validation = Array.prototype.filter.call(forms, function(form) {
      form.addEventListener('submit', function(event) {
        if (form.checkValidity() === false) {
          event.preventDefault();
          event.stopPropagation();
        }
      });
    });
  });
}

```

```

        form.classList.add('was-validated');
    }, false);
});
}, false);
})();
</script>

```

浏览器默认值

Not interested in custom validation feedback messages or writing JavaScript to change form behaviors? All good, you can use the browser defaults. Try submitting the form below. Depending on your browser and OS, you'll see a slightly different style of feedback.

While these feedback styles cannot be styled with CSS, you can still customize the feedback text through JavaScript.

First name

Last name

Username

@

City

State

Zip

☐ Agree to terms and conditions

Submit form

```

<form>
  <div class="form-row">
    <div class="col-md-4 mb-3">
      <label for="validationDefault01">First name</label>
      <input type="text" class="form-control" id="validationDefault01"
placeholder="First name" value="Mark" required>
    </div>
    <div class="col-md-4 mb-3">
      <label for="validationDefault02">Last name</label>
      <input type="text" class="form-control" id="validationDefault02"
placeholder="Last name" value="Otto" required>
    </div>
    <div class="col-md-4 mb-3">
      <label for="validationDefaultUsername">Username</label>
      <div class="input-group">
        <div class="input-group-prepend">
          <span class="input-group-text" id="inputGroupPrepend2">@</span>
        </div>
        <input type="text" class="form-control" id="validationDefaultUsername"
placeholder="Username" aria-describedby="inputGroupPrepend2" required>
      </div>
    </div>
  </div>
</div>

```

```

<div class="form-row">
  <div class="col-md-6 mb-3">
    <label for="validationDefault03">City</label>
    <input type="text" class="form-control" id="validationDefault03"
placeholder="City" required>
  </div>
  <div class="col-md-3 mb-3">
    <label for="validationDefault04">State</label>
    <input type="text" class="form-control" id="validationDefault04"
placeholder="State" required>
  </div>
  <div class="col-md-3 mb-3">
    <label for="validationDefault05">Zip</label>
    <input type="text" class="form-control" id="validationDefault05"
placeholder="Zip" required>
  </div>
</div>
<div class="form-group">
  <div class="form-check">
    <input class="form-check-input" type="checkbox" value="" id="invalidCheck2"
required>
    <label class="form-check-label" for="invalidCheck2">
      Agree to terms and conditions
    </label>
  </div>
</div>
<button class="btn btn-primary" type="submit">Submit form</button>
</form>

```

服务器端

我们建议使用客户端验证，但是如果您需要使用服务器端验证，则可以使用`.is-invalid`和`.is-valid`来表示无效和有效的表单字段。注意，`.invalid-feedback`这些类也支持。

First name

Mark

Looks good!

Last name

Otto

Looks good!

Username

@ Username

Please choose a username.

City

City

Please provide a valid city.

State

State

Please provide a valid state.

Zip

Zip

Please provide a valid zip.

☐ Agree to terms and conditions
 You must agree before submitting.

Submit form

```

<form>
  <div class="form-row">

```

```

<div class="col-md-4 mb-3">
  <label for="validationServer01">First name</label>
  <input type="text" class="form-control is-valid" id="validationServer01"
placeholder="First name" value="Mark" required>
  <div class="valid-feedback">
    Looks good!
  </div>
</div>
<div class="col-md-4 mb-3">
  <label for="validationServer02">Last name</label>
  <input type="text" class="form-control is-valid" id="validationServer02"
placeholder="Last name" value="Otto" required>
  <div class="valid-feedback">
    Looks good!
  </div>
</div>
<div class="col-md-4 mb-3">
  <label for="validationServerUsername">Username</label>
  <div class="input-group">
    <div class="input-group-prepend">
      <span class="input-group-text" id="inputGroupPrepend3">@</span>
    </div>
    <input type="text" class="form-control is-invalid"
id="validationServerUsername" placeholder="Username"
aria-describedby="inputGroupPrepend3" required>
    <div class="invalid-feedback">
      Please choose a username.
    </div>
  </div>
</div>
</div>
<div class="form-row">
  <div class="col-md-6 mb-3">
    <label for="validationServer03">City</label>
    <input type="text" class="form-control is-invalid" id="validationServer03"
placeholder="City" required>
    <div class="invalid-feedback">
      Please provide a valid city.
    </div>
  </div>
  <div class="col-md-3 mb-3">
    <label for="validationServer04">State</label>
    <input type="text" class="form-control is-invalid" id="validationServer04"
placeholder="State" required>
    <div class="invalid-feedback">
      Please provide a valid state.
    </div>
  </div>
  <div class="col-md-3 mb-3">

```

```

    <label for="validationServer05">Zip</label>
    <input type="text" class="form-control is-invalid" id="validationServer05"
placeholder="Zip" required>
    <div class="invalid-feedback">
      Please provide a valid zip.
    </div>
  </div>
</div>
<div class="form-group">
  <div class="form-check">
    <input class="form-check-input is-invalid" type="checkbox" value=""
id="invalidCheck3" required>
    <label class="form-check-label" for="invalidCheck3">
      Agree to terms and conditions
    </label>
    <div class="invalid-feedback">
      You must agree before submitting.
    </div>
  </div>
</div>
<button class="btn btn-primary" type="submit">Submit form</button>
</form>

```

支持元素

示例表单显示了其中的原生文本 `<input>` 定义，但也可自定义表单控件提供表单验证样式。

☐ Check this custom checkbox

Example invalid feedback text

☐ Toggle this custom radio

☐ Or toggle this other custom radio

More example invalid feedback text

Open this select menu

Example invalid custom select feedback

Choose file...

Browse

Example invalid custom file feedback

```

<form class="was-validated">
  <div class="custom-control custom-checkbox mb-3">
    <input type="checkbox" class="custom-control-input"
id="customControlValidation1" required>
    <label class="custom-control-label" for="customControlValidation1">Check this
custom checkbox</label>
    <div class="invalid-feedback">Example invalid feedback text</div>
  </div>

  <div class="custom-control custom-radio">

```

```

    <input type="radio" class="custom-control-input"
id="customControlValidation2" name="radio-stacked" required>
    <label class="custom-control-label" for="customControlValidation2">Toggle
this custom radio</label>
</div>
<div class="custom-control custom-radio mb-3">
    <input type="radio" class="custom-control-input"
id="customControlValidation3" name="radio-stacked" required>
    <label class="custom-control-label" for="customControlValidation3">Or toggle
this other custom radio</label>
    <div class="invalid-feedback">More example invalid feedback text</div>
</div>

<div class="form-group">
    <select class="custom-select" required>
        <option value="">Open this select menu</option>
        <option value="1">One</option>
        <option value="2">Two</option>
        <option value="3">Three</option>

    </select>
    <div class="invalid-feedback">Example invalid custom select feedback</div>
</div>

<div class="custom-file">
    <input type="file" class="custom-file-input" id="validatedCustomFile"
required>
    <label class="custom-file-label" for="validatedCustomFile">Choose
file...</label>
    <div class="invalid-feedback">Example invalid custom file feedback</div>
</div>
</form>

```

提示

如果表单布局允许，则可以交换。 `.{valid | invalid} --tooltip` 类的。 `.{valid|invalid}-tooltip` 以在样式化的工具提示中显示验证反馈。 确保有一个父母的位置： 相对工具提示定位。 在下面的例子中，我们的列类已经有了，但是你的项目可能需要一个替代的设置。

First name

Mark

Last name

Otto

Username

@ Username

City

City

State

State

Zip

Zip

Submit form


```

<form class="needs-validation" novalidate>
  <div class="form-row">
    <div class="col-md-4 mb-3">
      <label for="validationTooltip01">First name</label>
      <input type="text" class="form-control" id="validationTooltip01"
placeholder="First name" value="Mark" required>
      <div class="valid-tooltip">
        Looks good!
      </div>
    </div>
    <div class="col-md-4 mb-3">
      <label for="validationTooltip02">Last name</label>
      <input type="text" class="form-control" id="validationTooltip02"
placeholder="Last name" value="Otto" required>
      <div class="valid-tooltip">
        Looks good!
      </div>
    </div>
    <div class="col-md-4 mb-3">
      <label for="validationTooltipUsername">Username</label>
      <div class="input-group">
        <div class="input-group-prepend">
          <span class="input-group-text"
id="validationTooltipUsernamePrepend">@</span>
        </div>
        <input type="text" class="form-control" id="validationTooltipUsername"
placeholder="Username" aria-describedby="validationTooltipUsernamePrepend"
required>
        <div class="invalid-tooltip">
          Please choose a unique and valid username.
        </div>
      </div>
    </div>
  </div>
  <div class="form-row">
    <div class="col-md-6 mb-3">
      <label for="validationTooltip03">City</label>
      <input type="text" class="form-control" id="validationTooltip03"
placeholder="City" required>
      <div class="invalid-tooltip">
        Please provide a valid city.
      </div>
    </div>
    <div class="col-md-3 mb-3">
      <label for="validationTooltip04">State</label>
      <input type="text" class="form-control" id="validationTooltip04"
placeholder="State" required>
      <div class="invalid-tooltip">

```

```

        Please provide a valid state.
    </div>
</div>
<div class="col-md-3 mb-3">
    <label for="validationTooltip05">Zip</label>
    <input type="text" class="form-control" id="validationTooltip05"
placeholder="Zip" required>
    <div class="invalid-tooltip">
        Please provide a valid zip.
    </div>
</div>
</div>
<button class="btn btn-primary" type="submit">Submit form</button>
</form>

```

自定义表单

为了使自定义表单和跨浏览器保持一致性，请使用自定义的表单元素来替换浏览器的默认值，它们建立在语义和具备友好的标记之上，因此它们是可以替代任何默认表单控制元件的。

复选框和单选框自定义

每个选取块和选项按钮被包在一个 `<label>` 中，因为理由有三：

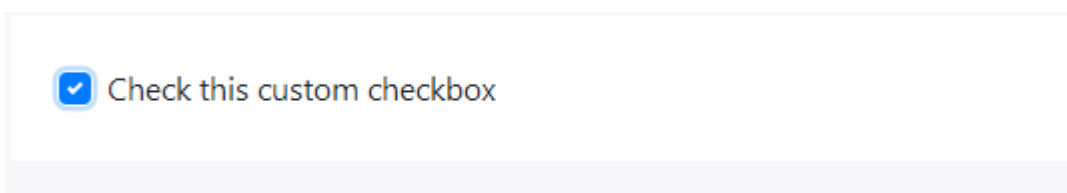
- 这样能提供更大的点击区域。
- 这样能提供一个更加有意义的语义容器，以帮助我们替换默认的 `<input>` 组件。
- 这样会自动触发 `<input>` 的状态，省去引用 JavaScript 插件。

我们使用 `opacity` 隐藏默认的 `<input>` 并使用 `.custom-control-indicator` 在它的位置之上构建一个全新的表单指示器，遗憾的是，由于 CSS 的 `content` 对该元素不起作用，所以我们不能仅使用 `<input>` 构建一个自定义事件。

我们使用 `(~)CSS` 兄弟选择器来为 `<input>` 处理样式和状态，就象 `:checked` 自定义窗体指示符一样，当与 `.custom-control-description` 结合使用，也可以根据 `<input>` 的状态对每个对象进行样式调整。

在检查状态下，我们使用 [Open Iconic](#) 的 **base64 embedded SVG icons** 图标，提供跨浏览器和样式设计的最佳控制器。

Checkbox 勾选



```

<div class="custom-control custom-checkbox">
    <input type="checkbox" class="custom-control-input" id="customCheck1">
    <label class="custom-control-label" for="customCheck1">Check this custom
checkbox</label>

```

</div>

通过 JavaScript(当没有可用的 HTML 属性来指定它) 手机设置时，自定义勾选控件还可以使用 `:indeterminate` 伪类（实现一种模糊的提示使用的样式），可见 bbs.z01.com/PItem?ID=15849

☒ Check this custom checkbox

如果你正使用 jQuery，可以用下面方法：

```
$('.your-checkbox').prop('indeterminate', true)
```

单选项

☐ Toggle this custom radio
☒ Or toggle this other custom radio

```
<div class="custom-control custom-radio">
  <input type="radio" id="customRadio1" name="customRadio"
class="custom-control-input">
  <label class="custom-control-label" for="customRadio1">Toggle this custom
radio</label>
</div>
<div class="custom-control custom-radio">
  <input type="radio" id="customRadio2" name="customRadio"
class="custom-control-input">
  <label class="custom-control-label" for="customRadio2">Or toggle this other
custom radio</label>
</div>
```

一致

☐ Toggle this custom radio ☐ Or toggle this other custom radio

```
<div class="custom-control custom-radio custom-control-inline">
  <input type="radio" id="customRadioInline1" name="customRadioInline1"
class="custom-control-input">
  <label class="custom-control-label" for="customRadioInline1">Toggle this
custom radio</label>
</div>
<div class="custom-control custom-radio custom-control-inline">
```

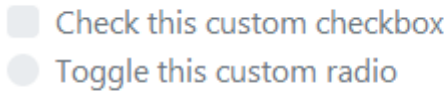
```

        <input type="radio" id="customRadioInline2" name="customRadioInline1"
class="custom-control-input">
        <label class="custom-control-label" for="customRadioInline2">Or toggle
this other custom radio</label>
    </div>

```

禁用

自定义单选框和复选框也可以禁用，只要将 `disabled` 布尔值属性加入到 `<input>`，禁用的指示和样式都会自动载入。



```

<div class="custom-control custom-checkbox">
    <input type="checkbox" class="custom-control-input" id="customCheckDisabled"
disabled>
    <label class="custom-control-label" for="customCheckDisabled">Check this custom
checkbox</label>
</div>

<div class="custom-control custom-radio">
    <input type="radio" id="radio3" name="radioDisabled" id="customRadioDisabled"
class="custom-control-input" disabled>
    <label class="custom-control-label" for="customRadioDisabled">Toggle this
custom radio</label>
</div>

```

堆叠

自訂的核取方塊和單選按鈕原本會以行內樣式排列。加入一個 `.custom-controls-stacked` 的父級，以確保每個表單控制元件在不同的列上。 自定义的单选框和复选框，原本会以行内样式水平排列，加入一个 `.custom-controls-stacked` 父级，则可以确保每个控件在不同的行上，形成堆叠效果。

```

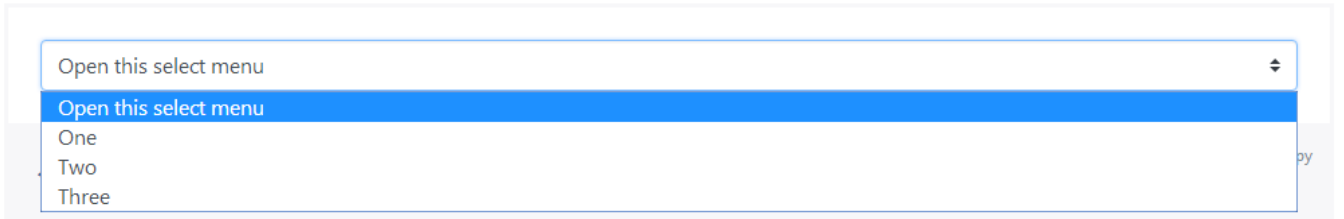
<div class="custom-controls-stacked">
    <label class="custom-control custom-radio">
        <input id="radioStacked3" name="radio-stacked" type="radio"
class="custom-control-input">
        <span class="custom-control-indicator"></span>
        <span class="custom-control-description">Toggle this custom radio</span>
    </label>
    <label class="custom-control custom-radio">
        <input id="radioStacked4" name="radio-stacked" type="radio"
class="custom-control-input">
        <span class="custom-control-indicator"></span>
        <span class="custom-control-description">Or toggle this other custom
radio</span>

```

```
</label>
</div>
```

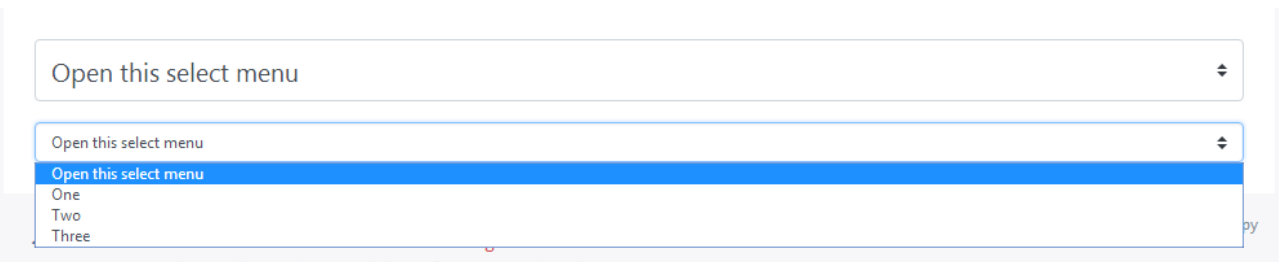
Select menu 下拉选择菜单

自定义<select>下拉选择菜单只需要一个.custom-selectCSS 即可触发自定义样式。



```
<select class="custom-select">
  <option selected>Open this select menu</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>
```

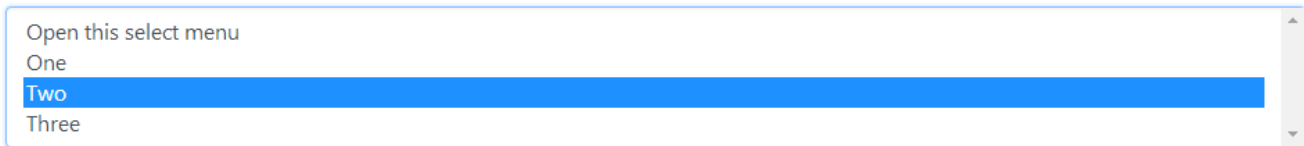
您也可以选择小和大的自定义选择来匹配我们相似大小的文本输入。



```
<select class="custom-select custom-select-lg mb-3">
  <option selected>Open this select menu</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>

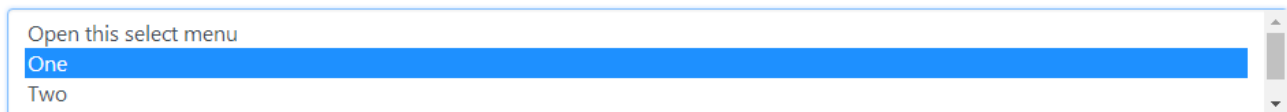
<select class="custom-select custom-select-sm">
  <option selected>Open this select menu</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>
```

多重属性也被支持:



```
<select class="custom-select" multiple>
  <option selected>Open this select menu</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>
```

如 `size` 属性:



```
<select class="custom-select" size="3">
  <option selected>Open this select menu</option>
  <option value="1">One</option>
  <option value="2">Two</option>
  <option value="3">Three</option>
</select>
```

文件浏览（文件选取）器

文件浏览（选取）是比较原始粗糙的，它需要额外的 JavaScript 定义支持，如果你将 *Choose file...* 文件选取和所选文件的名称关联。



```
<div class="custom-file">
  <input type="file" class="custom-file-input" id="customFile">
  <label class="custom-file-label" for="customFile">Choose file</label>
</div>
```

Here's how it works:

- We wrap the `<input>` in a `<label>` so the custom control properly triggers the file browser.
- We hide the default file `<input>` via `opacity`.
- We use `::after` to generate a custom background and directive (*Choose file...*).
- We use `::before` to generate and position the *Browse* button.
- We declare a `height` on the `<input>` for proper spacing for surrounding content.

In other words, it's an entirely custom element, all generated via CSS.

Translating or customizing the strings

The `:lang()` [pseudo-class](#) is used to allow for easy translation of the "Browse" and "Choose file..." text into other languages. Simply override or add entries to the `$custom-file-text` SCSS variable with the relevant [language tag](#) and localized strings. The English strings can be customized the same way. For example, here's how one might add a Spanish translation (Spanish's language code is `es`):

Copy

```
$custom-file-text: (  
  en: "Browse",  
  es: "Elegir"  
);
```

Here's `lang(es)` in action on the custom file input for a Spanish translation:

Seleccionar Archivo

Browse

```
<div class="custom-file">  
  <input type="file" class="custom-file-input" id="customFileLang" lang="es">  
  <label class="custom-file-label" for="customFileLang">Seleccionar  
Archivo</label>  
</div>
```

You'll need to set the language of your document (or subtree thereof) correctly in order for the correct text to be shown. This can be done using [the lang attribute](#) or the [Content-Language HTTP header](#), among other methods.

input 输入框及输入框群组 (Input-group)

在文本框的两侧定义文本、按钮或按钮组，轻松扩展表单控件。

基本示例

在 input 的任何一侧放置一个附加组件或按钮，您也可以在 Input 输入框的两面各放置一个组件（或按钮），我们不支持单个输入组合中有多个表单控制项，同时 `<label>` 必须在输入框组之外。

Four examples of Bootstrap input groups are shown:

- Basic:** An input field with the placeholder "Username" and an "@" symbol prepended.
- Append:** An input field with the placeholder "Recipient's username" and "@example.com" appended.
- Prepend:** An input field with the placeholder "Your vanity URL" and "https://example.com/users/" prepended.
- Both:** An input field with the placeholder "With textarea" (though the image shows a text input), with "\$" prepended and ".00" appended.

```
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text" id="basic-addon1">@</span>
  </div>
  <input type="text" class="form-control" placeholder="Username"
aria-label="Username" aria-describedby="basic-addon1">
</div>
```

```
<div class="input-group mb-3">
  <input type="text" class="form-control" placeholder="Recipient's username"
aria-label="Recipient's username" aria-describedby="basic-addon2">
  <div class="input-group-append">
    <span class="input-group-text" id="basic-addon2">@example.com</span>
  </div>
</div>
```

```
<label for="basic-url">Your vanity URL</label>
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text"
id="basic-addon3">https://example.com/users/</span>
  </div>
  <input type="text" class="form-control" id="basic-url"
aria-describedby="basic-addon3">
</div>
```



```

<div class="input-group mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text">$</span>
  </div>
  <input type="text" class="form-control" aria-label="Amount (to the nearest
dollar)">
  <div class="input-group-append">
    <span class="input-group-text">.<small>00</small></span>
  </div>
</div>

<div class="input-group">
  <div class="input-group-prepend">
    <span class="input-group-text">With textarea</span>
  </div>
  <textarea class="form-control" aria-label="With textarea"></textarea>
</div>

```

规格尺寸定义

将相对表单大小的 class 样式加到 `.input-group` 中，其内容会自动调整大小，如 `.input-group-lg`、`.input-group-sm`，不需要在每个元素上重重使用样式控制其大小。

Small

Default

Large

```

<div class="input-group input-group-sm mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text" id="inputGroup-sizing-sm">Small</span>
  </div>
  <input type="text" class="form-control" aria-label="Small"
aria-describedby="inputGroup-sizing-sm">
</div>

<div class="input-group mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text" id="inputGroup-sizing-default">Default</span>
  </div>
  <input type="text" class="form-control" aria-label="Default"
aria-describedby="inputGroup-sizing-default">
</div>

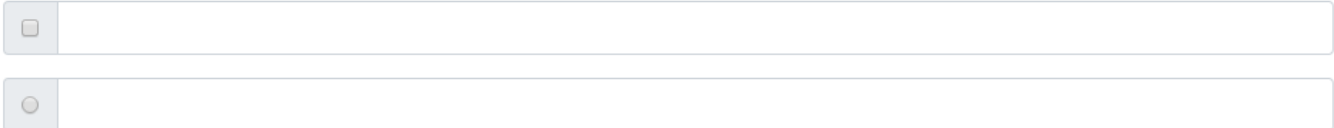
<div class="input-group input-group-lg">
  <div class="input-group-prepend">

```

```
    <span class="input-group-text" id="inputGroup-sizing-lg">Large</span>
  </div>
  <input type="text" class="form-control" aria-label="Large"
aria-describedby="inputGroup-sizing-sm">
</div>
```

勾选或单选框组合

在输入组的插件中放置任何复选框或单选七天，而不仅仅文本。

The image shows two examples of Bootstrap input groups. The first example features a small square checkbox icon inside a light gray box, followed by a text input field. The second example features a small circular radio button icon inside a light gray box, followed by a text input field.

```
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <div class="input-group-text">
      <input type="checkbox" aria-label="Checkbox for following text input">
    </div>
  </div>
  <input type="text" class="form-control" aria-label="Text input with checkbox">
</div>

<div class="input-group">
  <div class="input-group-prepend">
    <div class="input-group-text">
      <input type="radio" aria-label="Radio button for following text input">
    </div>
  </div>
  <input type="text" class="form-control" aria-label="Text input with radio
button">
</div>
```

多个输入

尽管可视化支持多个 `<input>`但验证样式仅适用于具有单个`<input>`的输入组。

The image shows a Bootstrap input group. It starts with a light gray box containing the text 'First and last name'. This is followed by two adjacent text input fields, each with a light gray border.

```
<div class="input-group">
  <div class="input-group-prepend">
    <span class="input-group-text" id="">First and last name</span>
  </div>
  <input type="text" class="form-control">
  <input type="text" class="form-control">
</div>
```

多类型控件组合

支持多种控件结合，比如复选框和、文本、input 框混合使用。

\$0.00

\$0.00

```
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text">$</span>
    <span class="input-group-text">0.00</span>
  </div>
  <input type="text" class="form-control" aria-label="Amount (to the nearest dollar)">
</div>

<div class="input-group">
  <input type="text" class="form-control" aria-label="Amount (to the nearest dollar)">
  <div class="input-group-append">
    <span class="input-group-text">$</span>
    <span class="input-group-text">0.00</span>
  </div>
</div>
```

按钮组合

Button

Button

Button

Button

Button

Button

```
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
  <input type="text" class="form-control" placeholder="" aria-label=""
  aria-describedby="basic-addon1">
</div>

<div class="input-group mb-3">
```

```

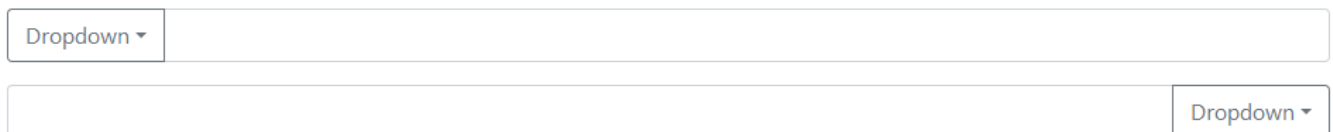
<input type="text" class="form-control" placeholder="Recipient's username"
aria-label="Recipient's username" aria-describedby="basic-addon2">
<div class="input-group-append">
  <button class="btn btn-outline-secondary" type="button">Button</button>
</div>
</div>

<div class="input-group mb-3">
  <div class="input-group-prepend">
    <button class="btn btn-outline-secondary" type="button">Button</button>
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
  <input type="text" class="form-control" placeholder="" aria-label=""
aria-describedby="basic-addon1">
</div>

<div class="input-group">
  <input type="text" class="form-control" placeholder="Recipient's username"
aria-label="Recipient's username" aria-describedby="basic-addon2">
  <div class="input-group-append">
    <button class="btn btn-outline-secondary" type="button">Button</button>
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
</div>

```

帶下拉列表的按鈕組合



The image displays two examples of Bootstrap input groups. The first example shows a dropdown button (labeled 'Dropdown') on the left of a text input. The second example shows a text input on the left of a dropdown button (labeled 'Dropdown').

```

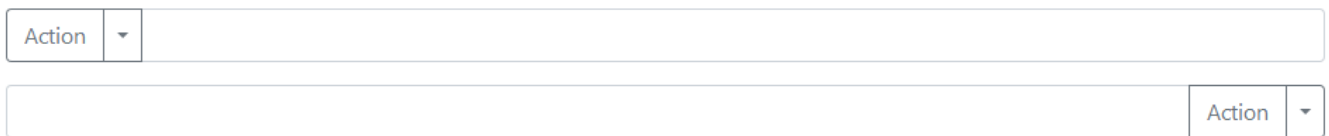
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <button class="btn btn-outline-secondary dropdown-toggle" type="button"
data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">Dropdown</button>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Action</a>
      <a class="dropdown-item" href="#">Another action</a>
      <a class="dropdown-item" href="#">Something else here</a>
      <div role="separator" class="dropdown-divider"></div>
      <a class="dropdown-item" href="#">Separated link</a>
    </div>
  </div>
  <input type="text" class="form-control" aria-label="Text input with dropdown
button">

```

```
</div>
```

```
<div class="input-group">
  <input type="text" class="form-control" aria-label="Text input with dropdown
button">
  <div class="input-group-append">
    <button class="btn btn-outline-secondary dropdown-toggle" type="button"
data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">Dropdown</button>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Action</a>
      <a class="dropdown-item" href="#">Another action</a>
      <a class="dropdown-item" href="#">Something else here</a>
      <div role="separator" class="dropdown-divider"></div>
      <a class="dropdown-item" href="#">Separated link</a>
    </div>
  </div>
</div>
```

分裂式按钮与 input 组合



```
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <button type="button" class="btn btn-outline-secondary">Action</button>
    <button type="button" class="btn btn-outline-secondary dropdown-toggle
dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
      <span class="sr-only">Toggle Dropdown</span>
    </button>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Action</a>
      <a class="dropdown-item" href="#">Another action</a>
      <a class="dropdown-item" href="#">Something else here</a>
      <div role="separator" class="dropdown-divider"></div>
      <a class="dropdown-item" href="#">Separated link</a>
    </div>
  </div>
  <input type="text" class="form-control" aria-label="Text input with segmented
dropdown button">
</div>

<div class="input-group">
```

```
<input type="text" class="form-control" aria-label="Text input with segmented
dropdown button">
<div class="input-group-append">
  <button type="button" class="btn btn-outline-secondary">Action</button>
  <button type="button" class="btn btn-outline-secondary dropdown-toggle
dropdown-toggle-split" data-toggle="dropdown" aria-haspopup="true"
aria-expanded="false">
    <span class="sr-only">Toggle Dropdown</span>
  </button>
  <div class="dropdown-menu">
    <a class="dropdown-item" href="#">Action</a>
    <a class="dropdown-item" href="#">Another action</a>
    <a class="dropdown-item" href="#">Something else here</a>
    <div role="separator" class="dropdown-divider"></div>
    <a class="dropdown-item" href="#">Separated link</a>
  </div>
</div>
</div>
```

自定义表单

输入组包括对自定义选择和自定义文件输入的支持。 这些浏览器的默认版本不受支持。

自定义选择

Options

Choose...

Choose...

Options

Button

Choose...

Choose...

Button

```
<div class="input-group mb-3">
  <div class="input-group-prepend">
    <label class="input-group-text" for="inputGroupSelect01">Options</label>
  </div>
  <select class="custom-select" id="inputGroupSelect01">
    <option selected>Choose...</option>
    <option value="1">One</option>
    <option value="2">Two</option>
    <option value="3">Three</option>
  </select>
</div>

<div class="input-group mb-3">
  <select class="custom-select" id="inputGroupSelect02">
```

```

    <option selected>Choose...</option>
    <option value="1">One</option>
    <option value="2">Two</option>
    <option value="3">Three</option>
  </select>
  <div class="input-group-append">
    <label class="input-group-text" for="inputGroupSelect02">Options</label>
  </div>
</div>

<div class="input-group mb-3">
  <div class="input-group-prepend">
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
  <select class="custom-select" id="inputGroupSelect03">
    <option selected>Choose...</option>
    <option value="1">One</option>
    <option value="2">Two</option>
    <option value="3">Three</option>
  </select>
</div>

<div class="input-group">
  <select class="custom-select" id="inputGroupSelect04">
    <option selected>Choose...</option>
    <option value="1">One</option>
    <option value="2">Two</option>
    <option value="3">Three</option>
  </select>
  <div class="input-group-append">
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
</div>

```

自定义文件输入

Upload

Choose file

Browse

Choose file

Browse

Upload

Button

Choose file

Browse

Choose file

Browse

Button

```

<div class="input-group mb-3">
  <div class="input-group-prepend">
    <span class="input-group-text">Upload</span>

```

```

</div>
<div class="custom-file">
  <input type="file" class="custom-file-input" id="inputGroupFile01">
  <label class="custom-file-label" for="inputGroupFile01">Choose file</label>
</div>
</div>

<div class="input-group mb-3">
  <div class="custom-file">
    <input type="file" class="custom-file-input" id="inputGroupFile02">
    <label class="custom-file-label" for="inputGroupFile02">Choose file</label>
  </div>
  <div class="input-group-append">
    <span class="input-group-text" id="">Upload</span>
  </div>
</div>

<div class="input-group mb-3">
  <div class="input-group-prepend">
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
  <div class="custom-file">
    <input type="file" class="custom-file-input" id="inputGroupFile03">
    <label class="custom-file-label" for="inputGroupFile03">Choose file</label>
  </div>
</div>

<div class="input-group">
  <div class="custom-file">
    <input type="file" class="custom-file-input" id="inputGroupFile04">
    <label class="custom-file-label" for="inputGroupFile04">Choose file</label>
  </div>
  <div class="input-group-append">
    <button class="btn btn-outline-secondary" type="button">Button</button>
  </div>
</div>

```

无障碍浏览提示(易用性)

如果您没有为每个 `input` 输入添加 `<label>`，屏幕阅读器将对您的表单有困难。对于这些输入组，请确保向辅助技术传达任何附加的标签或功能。

要使用的确切技术（`<label>`使用 `.sr-only` 类隐藏的元素，或使用 `aria-labelledby` 和 `aria-describedby` 属性结合 `aria-describedby` 使用）以及需要传达的其他信息将根据您正在实现的界面小部件的具体类型而有所不同。本节中的示例提供了一些建议的具体案例。

Hero 广告大块屏幕(Jumbotron)

通过引用 `.jumbotron` 方法，实现构建超大的 Hero 界面，醒目的在网站上展示关键的营销信息，轻量、灵活。

这是一个轻量、灵活的组件，可以选择性地扩展整个视口，以展示您网站上的关键营销信息。

Hello, world!

This is a simple hero unit, a simple jumbotron-style component for calling extra attention to featured content or information.

It uses utility classes for typography and spacing to space content out within the larger container.

[Learn more](#)

```
<div class="jumbotron">
  <h1 class="display-3">Hello, world!</h1>
  <p class="lead">This is a simple hero unit, a simple jumbotron-style component
for calling extra attention to featured content or information.</p>
  <hr class="my-4">
  <p>It uses utility classes for typography and spacing to space content out within
the larger container.</p>
  <p class="lead">
    <a class="btn btn-primary btn-lg" href="#" role="button">Learn more</a>
  </p>
</div>
```

想要让广告大屏幕占满当前显示浏览器全屏、不带有圆角，只要添加 `.jumbotron-fluid` CSS 修饰符，并在里面添加一个 `.container` 或 `.container-fluid` 内容空间即可。

Fluid jumbotron

This is a modified jumbotron that occupies the entire horizontal space of its parent.

```
<div class="jumbotron jumbotron-fluid">
  <div class="container">
    <h1 class="display-4">Fluid jumbotron</h1>
```

```
<p class="lead">This is a modified jumbotron that occupies the entire horizontal  
space of its parent.</p>  
</div>  
</div>
```

列表组 (List-group)

列表组是一个灵活而且强大的组件，不仅仅可以用来显示简单的元素列表，还可以通过定义显示复杂的内容。

基本示例

最基本的列表组就是一个无序列表、带有几个列表项以及适当的样式定义类。在此基础上可以使用下列选项，或根据你自己的需要调整 CSS。

Cras justo odio
Dapibus ac facilisis in
Morbi leo risus
Porta ac consectetur ac
Vestibulum at eros

```
<ul class="list-group">
  <li class="list-group-item">Cras justo odio</li>
  <li class="list-group-item">Dapibus ac facilisis in</li>
  <li class="list-group-item">Morbi leo risus</li>
  <li class="list-group-item">Porta ac consectetur ac</li>
  <li class="list-group-item">Vestibulum at eros</li>
</ul>
```

.active 列表(启用)状态指示

添加 `.active` 到 `.list-group-item` 下的其中一行或多行，以指示当前有效的选择状态。

Cras justo odio
Dapibus ac facilisis in
Morbi leo risus
Porta ac consectetur ac
Vestibulum at eros

```
<ul class="list-group">
  <li class="list-group-item active">Cras justo odio</li>
  <li class="list-group-item">Dapibus ac facilisis in</li>
  <li class="list-group-item">Morbi leo risus</li>
  <li class="list-group-item">Porta ac consectetur ac</li>
  <li class="list-group-item">Vestibulum at eros</li>
</ul>
```

.disabled 列表(禁用)状态指示

添加 `.disabled` 到 `.list-group-item` 下的其中一行或多行，以显示出 禁用 状态。注意：一些带有 `.disabled` 的元素还需要自定义 JavaScript 脚本才能完全禁用其点击事件（如链接）。

Cras justo odio
Dapibus ac facilisis in
Morbi leo risus
Porta ac consectetur ac
Vestibulum at eros

```
<ul class="list-group">
  <li class="list-group-item disabled">Cras justo odio</li>
  <li class="list-group-item">Dapibus ac facilisis in</li>
  <li class="list-group-item">Morbi leo risus</li>
  <li class="list-group-item">Porta ac consectetur ac</li>
  <li class="list-group-item">Vestibulum at eros</li>
</ul>
```

链接和按钮

使用 `<a>`或 `<button>`来创建具有 hover、禁用、悬停和活动状态的列表组 `.list-group-item-action`，我们分离这些 Class 样式，以确保由非交互元素组成的列表群组(如 ``或 `<div>`)不提供可点击或触击（即可以用一个父 `<div>`代替 ``-译者注）。

请务必 不要在这里使用 `.btn` 标准样式。

Cras justo odio
Dapibus ac facilisis in
Morbi leo risus
Porta ac consectetur ac
Vestibulum at eros

```
<div class="list-group">
  <a href="#" class="list-group-item list-group-item-action active">
    Cras justo odio
  </a>
  <a href="#" class="list-group-item list-group-item-action">Dapibus ac facilisis
in</a>
  <a href="#" class="list-group-item list-group-item-action">Morbi leo risus</a>
  <a href="#" class="list-group-item list-group-item-action">Porta ac consectetur
ac</a>
  <a href="#" class="list-group-item list-group-item-action disabled">Vestibulum
at eros</a>
</div>
```

使用<button>元素，你还可以使用 `disabled` 属性来达到禁用状态指示（或引用 `.disabled` 样式），不过这一属性不支持 HTML5 中的 `<a>` 标签。

Cras justo odio
Dapibus ac facilisis in
Morbi leo risus
Porta ac consectetur ac
Vestibulum at eros

```
<div class="list-group">
  <button type="button" class="list-group-item list-group-item-action active">
    Cras justo odio
  </button>
  <button type="button" class="list-group-item list-group-item-action">Dapibus ac
facilisis in</button>
```

```
<button type="button" class="list-group-item list-group-item-action">Morbi leo  
risus</button>  
<button type="button" class="list-group-item list-group-item-action">Porta ac  
consectetur ac</button>  
<button type="button" class="list-group-item list-group-item-action"  
disabled>Vestibulum at eros</button>  
</div>
```

Flush 紧致贴齐

加入`.list-group-flush`选择器，可以实现移除部分边框以及圆角，从而产生边缘贴齐的列表组，这在诸如 Card 卡片等容器中很实用（达成更好的呈现效果）。

Cras justo odio

Dapibus ac facilisis in

Morbi leo risus

Porta ac consectetur ac

Vestibulum at eros

```
<ul class="list-group list-group-flush">  
  <li class="list-group-item">Cras justo odio</li>  
  <li class="list-group-item">Dapibus ac facilisis in</li>  
  <li class="list-group-item">Morbi leo risus</li>  
  <li class="list-group-item">Porta ac consectetur ac</li>  
  <li class="list-group-item">Vestibulum at eros</li>  
</ul>
```

上下文语境颜色呈现样式

使用情景式 class 样式来设计具有状态背景和颜色的列表组，呈现默认或链接状态。

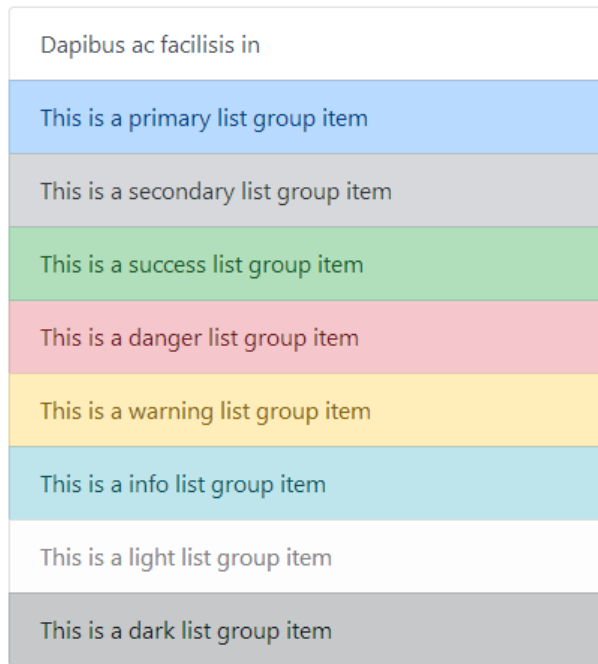
Dapibus ac facilisis in
This is a primary list group item
This is a secondary list group item
This is a success list group item
This is a danger list group item
This is a warning list group item
This is a info list group item
This is a light list group item
This is a dark list group item

```

<ul class="list-group">
  <li class="list-group-item">Dapibus ac facilisis in</li>
  <li class="list-group-item list-group-item-primary">This is a primary list group
item</li>
  <li class="list-group-item list-group-item-secondary">This is a secondary list
group item</li>
  <li class="list-group-item list-group-item-success">This is a success list group
item</li>
  <li class="list-group-item list-group-item-danger">This is a danger list group
item</li>
  <li class="list-group-item list-group-item-warning">This is a warning list group
item</li>
  <li class="list-group-item list-group-item-info">This is a info list group
item</li>
  <li class="list-group-item list-group-item-light">This is a light list group
item</li>
  <li class="list-group-item list-group-item-dark">This is a dark list group
item</li>
</ul>

```

情景式 class 也可以使用 `.list-group-item-action` 样式，注意，在上述示例中，不存在 hover 样式指示器，事实上它是支持的，而且还支持 `.active` 状态指示--我们可以应用它们在上下文情景列表组的项目上进行活动状态选择指示。



```
<div class="list-group">
  <a href="#" class="list-group-item list-group-item-action">Dapibus ac facilisis
in</a>
```

```
  <a href="#" class="list-group-item list-group-item-action
list-group-item-primary">This is a primary list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-secondary">This is a secondary list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-success">This is a success list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-danger">This is a danger list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-warning">This is a warning list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-info">This is a info list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-light">This is a light list group item</a>
  <a href="#" class="list-group-item list-group-item-action
list-group-item-dark">This is a dark list group item</a>
</div>
```

传达辅助技术的意义（易用性指南）

通过替代方法包括，例如隐藏在.sr-only 该类中的附加文本。 使用颜色添加意义只是提供一个视觉指示，这不会传达给使用辅助技术（如屏幕阅读器）的用户，推荐由颜色表达的内容从文字本身就显而易见，或者通过添加.sr-only class 选择器这样的替代方法来隐藏此类附加文本。

引入 badge 徽章

在[通用样式定义](#)的帮助下，可向任何列表项目添加 `.badge` 标签以显示未读计数、活动状态等。

Cras justo odio	14
Dapibus ac facilisis in	2
Morbi leo risus	1

```
<ul class="list-group">
  <li class="list-group-item d-flex justify-content-between align-items-center">
    Cras justo odio
    <span class="badge badge-primary badge-pill">14</span>
  </li>
  <li class="list-group-item d-flex justify-content-between align-items-center">
    Dapibus ac facilisis in
    <span class="badge badge-primary badge-pill">2</span>
  </li>
  <li class="list-group-item d-flex justify-content-between align-items-center">
    Morbi leo risus
    <span class="badge badge-primary badge-pill">1</span>
  </li>
</ul>
```

自定义内容

在 [flexbox 通用样式定义](#)的支持下，列表组中几乎可以添加任意的 HTML 内容，包括标签、内容、链接。

List group item heading3 days agoDonec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit. Donec id elit non mi porta.

List group item heading3 days agoDonec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit. Donec id elit non mi porta.

List group item heading3 days agoDonec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit. Donec id elit non mi porta.

```

<div class="list-group">
  <a href="#" class="list-group-item list-group-item-action flex-column align-items-start active">
    <div class="d-flex w-100 justify-content-between">
      <h5 class="mb-1">List group item heading</h5>
      <small>3 days ago</small>
    </div>
    <p class="mb-1">Donec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit.</p>
    <small>Donec id elit non mi porta.</small>
  </a>
  <a href="#" class="list-group-item list-group-item-action flex-column align-items-start">
    <div class="d-flex w-100 justify-content-between">
      <h5 class="mb-1">List group item heading</h5>
      <small class="text-muted">3 days ago</small>
    </div>
    <p class="mb-1">Donec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit.</p>
    <small class="text-muted">Donec id elit non mi porta.</small>
  </a>
  <a href="#" class="list-group-item list-group-item-action flex-column align-items-start">
    <div class="d-flex w-100 justify-content-between">
      <h5 class="mb-1">List group item heading</h5>
      <small class="text-muted">3 days ago</small>
    </div>
    <p class="mb-1">Donec id elit non mi porta gravida at eget metus. Maecenas sed diam eget risus varius blandit.</p>
    <small class="text-muted">Donec id elit non mi porta.</small>
  </a>
</div>

```

JavaScript 行为

使用列表组的 JavaScript 插件,单独或编译 `bootstrap.js` 文件来扩展我们的列表组，以提供可选择的内容列表，如下例：

Home
Profile
Messages
Settings

Ut ut do pariatu r aliquip aliqua aliquip exercitation do nostrud commodo reprehenderit aute ipsum voluptate. Irure Lorem et laboris nostrud amet cupidatat cupidatat anim do ut velit mollit consequat enim tempor. Consectetur est minim nostrud nostrud consectetur irure labore voluptate irure. Ipsum id Lorem sit sint voluptate est pariatu r eu ad cupidatat et deserunt culpa sit eiusmod deserunt. Consectetur et fugiat anim do eiusmod aliquip nulla laborum elit adipisicing pariatu r cillum.

```

<div class="row">
  <div class="col-4">
    <div class="list-group" id="list-tab" role="tablist">

```

```

        <a class="list-group-item list-group-item-action active" id="list-home-list"
data-toggle="list" href="#list-home" role="tab" aria-controls="home">Home</a>
        <a class="list-group-item list-group-item-action" id="list-profile-list"
data-toggle="list" href="#list-profile" role="tab"
aria-controls="profile">Profile</a>
        <a class="list-group-item list-group-item-action" id="list-messages-list"
data-toggle="list" href="#list-messages" role="tab"
aria-controls="messages">Messages</a>
        <a class="list-group-item list-group-item-action" id="list-settings-list"
data-toggle="list" href="#list-settings" role="tab"
aria-controls="settings">Settings</a>
    </div>
</div>
<div class="col-8">
    <div class="tab-content" id="nav-tabContent">
        <div class="tab-pane fade show active" id="list-home" role="tabpanel"
aria-labelledby="list-home-list">...</div>
        <div class="tab-pane fade" id="list-profile" role="tabpanel"
aria-labelledby="list-profile-list">...</div>
        <div class="tab-pane fade" id="list-messages" role="tabpanel"
aria-labelledby="list-messages-list">...</div>
        <div class="tab-pane fade" id="list-settings" role="tabpanel"
aria-labelledby="list-settings-list">...</div>
    </div>
</div>
</div>

```

使用数据属性

.list-group-item 上使用数据属性, 可以指定 `data-toggle="list"` 或在元素上编写任意的 JavaScript 事件来激活列表组导航, 而需要编写任何的 JavaScript 脚本。

Copy

```

<!-- List group -->
<div class="list-group" id="myList" role="tablist">
    <a class="list-group-item list-group-item-action active" data-toggle="list"
href="#home" role="tab">Home</a>
    <a class="list-group-item list-group-item-action" data-toggle="list"
href="#profile" role="tab">Profile</a>
    <a class="list-group-item list-group-item-action" data-toggle="list"
href="#messages" role="tab">Messages</a>
    <a class="list-group-item list-group-item-action" data-toggle="list"
href="#settings" role="tab">Settings</a>
</div>

<!-- Tab panes -->
<div class="tab-content">
    <div class="tab-pane active" id="home" role="tabpanel">...</div>

```

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

通过 JavaScript

通过 JavaScript 启用可选列表项（每个列表项需要单独激活）：

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

您可以通过以下几种方式激活单个列表项：

```
$('#myList a[href="#profile"]').tab('show') // Select tab by name
$('#myList a:first-child').tab('show') // Select first tab
$('#myList a:last-child').tab('show') // Select last tab
$('#myList a:nth-child(3)').tab('show') // Select third tab
```

fade 淡入淡出效果

要使定位的元素有淡入淡出效果，请将 `.fade` 添加到每个 `.tab-pane` 子项中，且第一个列表项目定义 `.show` 样式使之初始可见。

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

方法

`$.tab`

激活列表项元素和内容容器。标签应该具有 `data-target` 或 `href` 定位在 DOM 中的容器节点。

```
<div class="list-group" id="myList" role="tablist">
  <a class="list-group-item list-group-item-action active" data-toggle="list"
href="#home" role="tab">Home</a>
  <a class="list-group-item list-group-item-action" data-toggle="list"
href="#profile" role="tab">Profile</a>
  <a class="list-group-item list-group-item-action" data-toggle="list"
href="#messages" role="tab">Messages</a>
```

```
<a class="list-group-item list-group-item-action" data-toggle="list"
href="#settings" role="tab">Settings</a>
</div>

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

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

.tab('show')
```

选择给定的列表项并显示其关联的网页，之前选择的其它列表项目将被取消选中，并使其关联的表单空格隐藏，在选项卡实际显示前（如 `shown.bs.tab` 事件发生前）返回给调用者。

Copy
\$('#someListItem').tab('show')

Events 事件

当显示新标签时，事件按以下顺序触发：

1. `hide.bs.tab` (在当前活动的选项卡标签上)
2. `show.bs.tab` (在待显示的选项卡标签上)
3. `hidden.bs.tab` 在上一个活动的选项卡标签上，与 `hide.bs.tab` 事件相同)
4. `shown.bs.tab` (在新活动的、刚刚显示的选项卡标签上，与 `show.bs.tab` 事件相同)

如果没有活跃的选项卡标签，则 `hide.bs.tab` 与 `hidden.bs.tab` 事件不会被注销。

事件类型	描述
show.bs.tab	此事件在标签显示时触发，但在新标签显示之前。使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 将目前与此前启用（如果可用）的作为目标定位。
shown.bs.tab	此事件发生在选项卡显示后、新选项卡被显示之前，使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 分别定位启用中的选项卡和上一个活动选项卡（如果可用）。
hide.bs.tab	当要显示新的选项卡（因此先前的活动选项卡将被隐藏）时，此事件将触发。分别使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 定位当前活动选项卡 and 新的即将启用的选项卡。
hidden.bs.tab	在显示新标签页之后触发此事件（因此先前的活动标签页被隐藏），使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 分别定位上一个活动选项卡 and 新的活动选项卡。

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

.nav 导航/滑动门(nav)

基本导航样式

Bootstrap 中提供的导航可共享通用标记和样式，从基础 .nav 样式类到活动与禁用状态。交换 class 选择符以在每种样式之间切换。

基础 .nav 组件采用 Flexbox 弹性布局构建，并为构建所有类型的导航组件提供了坚实的基础，包括一些风格覆盖（以及列表）、一些更大 padding 连接间隙和基本的禁用样式。

基础的 .nav 组件不包含任何的 .active 状态，以下范例包括该类别，主要是为了说明这个 class 不会触发任何特殊的样式。

Active Link Link Disabled

```
<ul class="nav">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

.nav 的 class 可以使用在很多地方，所以你的标记可以非常灵活，比如使用在 列表，或者自定义一个 <nav>组件，因为 .nav 基于 display: flex 定义，导航链接的行为与导航项目相同，不需要额外的标记。

Active Link Link Disabled

```
<nav class="nav">
  <a class="nav-link active" href="#">Active</a>
  <a class="nav-link" href="#">Link</a>
  <a class="nav-link" href="#">Link</a>
  <a class="nav-link disabled" href="#">Disabled</a>
</nav>
```

可用样式

使用通用样式定义 `.nav` 元件，根据需要进行混合搭配、或自行建立。

水平对齐

使用 [flexbox 通用布局属性](#)更改导航的水平对齐方式，默认情况下，导航按左对齐，但你可以轻松将其改为居中对齐（或右对齐）。

用 `.justify-content-center` 居中

Active Link Link Disabled

```
<ul class="nav justify-content-center">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

使用 `.justify-content-end` 右对齐：

Active Link Link Disabled

```
<ul class="nav justify-content-end">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

垂直排列

使用 `.flex-column` 通用样式更改 Flexa 弹性项目的方向来设计导航，如在特定的 viewport 屏幕下需要堆叠，可使用响应式定义（如 `.flex-sm-column` 样式）。

Active

Link

Link

Disabled

```
<ul class="nav flex-column">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

和往常一样，垂直导航也可以没有 `` 选择器。

Active

Link

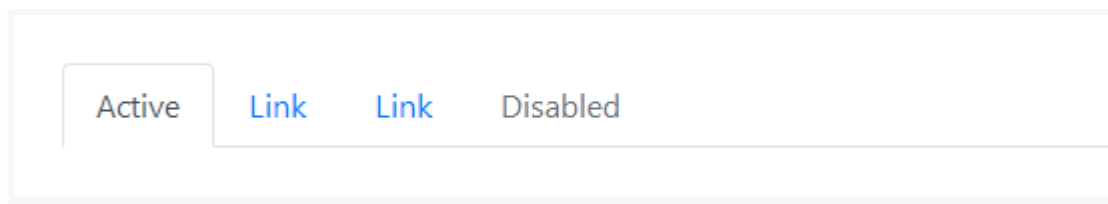
Link

Disabled

```
<nav class="nav flex-column">
  <a class="nav-link active" href="#">Active</a>
  <a class="nav-link" href="#">Link</a>
  <a class="nav-link" href="#">Link</a>
  <a class="nav-link disabled" href="#">Disabled</a>
</nav>
```

Tabs 标签

从上面了解的基本导航，并加入 `.nav-tabs` 以生成选项卡标签（滑动门，同时结合 [tab JavaScript 插件](#)来构建 tabs 滑动门。



```
<ul class="nav nav-tabs">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

胶囊式标签页

HTML 标记相同，但使用 `.nav-pills` 替代



```
<ul class="nav nav-pills">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

填充和对齐

`.nav` 内容有两种扩展 `class` 属性，使用 `.nav-fill` 会将 `.nav-item` 按照比例分配空间。注意：这会占用所有的水平空间，但不是每个导航项目的宽度相同。



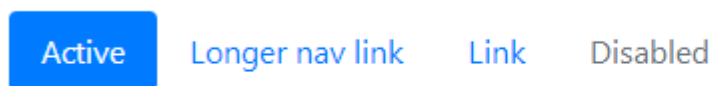
```
<ul class="nav nav-pills nav-fill">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Longer nav link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

当使用 `<nav>` 基于导航时，请确保包含 `.nav-item` 在 A 链接上。



```
<nav class="nav nav-pills nav-fill">
  <a class="nav-item nav-link active" href="#">Active</a>
  <a class="nav-item nav-link" href="#">Link</a>
  <a class="nav-item nav-link" href="#">Link</a>
  <a class="nav-item nav-link disabled" href="#">Disabled</a>
</nav>
```

对于等宽元素，请使用 `.nav-justified`。所有水平空间将被导航链接占用，但与上述 `.nav-fill` 不同，每个导航项目将具有相同的宽度。



```
<nav class="nav nav-pills nav-justified">
  <a class="nav-link active" href="#">Active</a>
  <a class="nav-link" href="#">Longer nav link</a>
  <a class="nav-link" href="#">Link</a>
  <a class="nav-link disabled" href="#">Disabled</a>
</nav>
```

与 `.nav-fill` 的例子近似，使用基于 `<nav>` 的导航，确保在链接上包含 `.nav-item` 子项定义。

Active

Link

Link

Disabled

```
<nav class="nav nav-pills nav-justified">
  <a class="nav-item nav-link active" href="#">Active</a>
  <a class="nav-item nav-link" href="#">Link</a>
  <a class="nav-item nav-link" href="#">Link</a>
  <a class="nav-item nav-link disabled" href="#">Disabled</a>
</nav>
```

使用 Flex 弹性布局

如果需要吊牌应式的导航变化，请使用一系列的 [flexbox 弹性布局类别](#)，这结通用类别能提供不同的弹性布局，下面示例中，我们的导及将会堆叠在最小的屏幕标准上，然后从小断点开始填充可用宽度的水平布局。

Active

Link

Link

Disabled

```
<nav class="nav nav-pills flex-column flex-sm-row">
  <a class="flex-sm-fill text-sm-center nav-link active" href="#">Active</a>
  <a class="flex-sm-fill text-sm-center nav-link" href="#">Link</a>
  <a class="flex-sm-fill text-sm-center nav-link" href="#">Link</a>
  <a class="flex-sm-fill text-sm-center nav-link disabled" href="#">Disabled</a>
</nav>
```

无障碍易用性处理

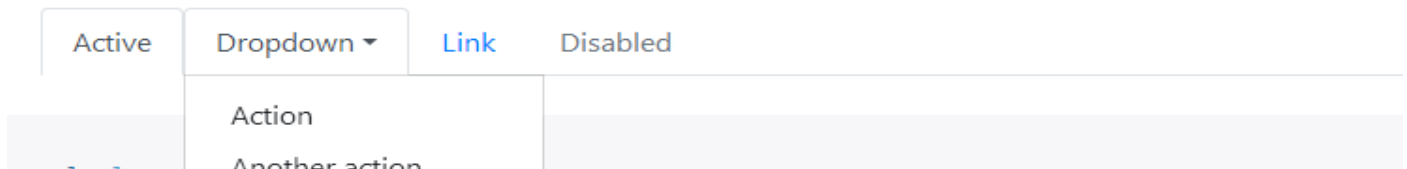
如果您正在使用导航栏提供导航栏，请确保添加 `a role="navigation"` 到最合理的 `` 父容器，或者 `<nav>` 围绕整个导航包装元素。不要为 `` 自己添加角色，因为这将阻止它被辅助技术公布为实际列表。

请注意，导航栏，即使视觉样式为 `.nav-tabs` 类的选项卡，也不应该给出 `role="tablist"`、`role="tab"` 或 `role="tabpanel"` 属性。这些仅适用于动态标签界面，如 WAI [ARIA 创作实践](#) 中所述。有关示例，请参阅本节中的 [动态选项卡界面的 JavaScript](#) 章节。

使用下拉菜单

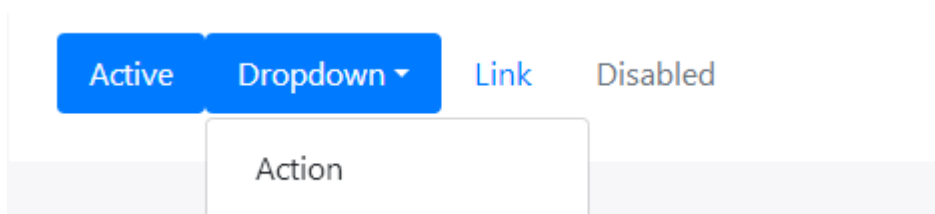
添加一些额外的 HTML 和 [下拉列表 JavaScript 插件](#) 的下拉菜单。

带下拉列表的标签页



```
<ul class="nav nav-tabs">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item dropdown">
    <a class="nav-link dropdown-toggle" data-toggle="dropdown" href="#"
role="button" aria-haspopup="true" aria-expanded="false">Dropdown</a>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Action</a>
      <a class="dropdown-item" href="#">Another action</a>
      <a class="dropdown-item" href="#">Something else here</a>
      <div class="dropdown-divider"></div>
      <a class="dropdown-item" href="#">Separated link</a>
    </div>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

帶下拉列表的膠囊式標籤頁



```
<ul class="nav nav-pills">
  <li class="nav-item">
    <a class="nav-link active" href="#">Active</a>
  </li>
  <li class="nav-item dropdown">
    <a class="nav-link dropdown-toggle" data-toggle="dropdown" href="#"
role="button" aria-haspopup="true" aria-expanded="false">Dropdown</a>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Action</a>
      <a class="dropdown-item" href="#">Another action</a>
      <a class="dropdown-item" href="#">Something else here</a>
      <div class="dropdown-divider"></div>
    </div>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

```

        <a class="dropdown-item" href="#">Separated link</a>
    </div>
</li>
<li class="nav-item">
    <a class="nav-link" href="#">Link</a>
</li>
<li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
</li>
</ul>

```

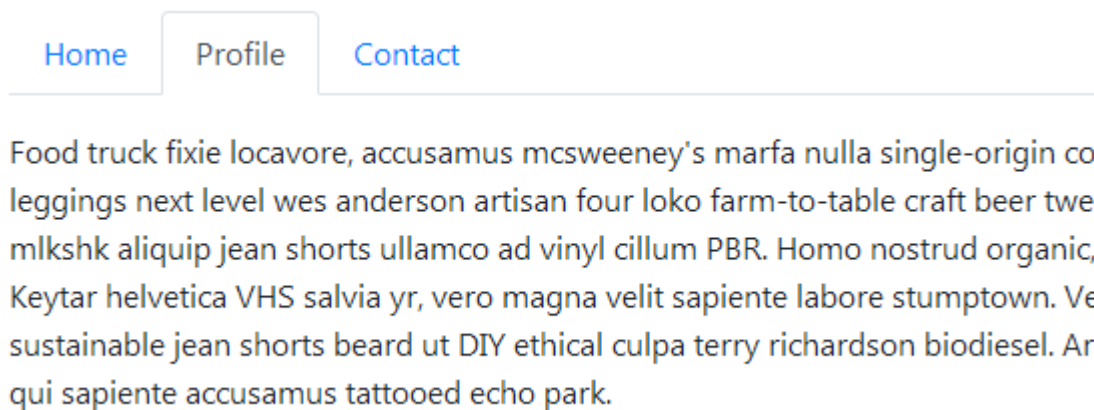
JavaScript 行为/滑动门

使用 JavaScript 标签页插件，单元包含或通过编译 `bootstrap.js` 文件来扩展我们的标签和按钮，以创建可选的选项卡、甚至是下拉菜单。

如果你需要从源码构建 JS，则需要 [包含 util.js](#) 进行编译。

动态标签的接口，如在所描述的 [WAI ARIA 创作实践](#) 创作实践，需要 `role="tablist"`、`role="tab"`、`role="tabpanel"`、和附加 `aria-` 属性，从而向用户提供辅助技术（如屏幕阅读器）结构、功能及状态反馈。

注意动态选项卡式界面不包含下拉菜单，因为这会导致可用性和可访问性问题。从可用性的角度来看，当前显示的选项卡的触发器元素不会立即可见（因为它在封闭的下拉菜单中）可能会导致混淆。从可访问性的角度来看，目前还没有明确的方式将这种结构映射到标准的 WAI ARIA 模式，这意味着它不能容易地被用户辅助技术所理解。



```

<ul class="nav nav-tabs" id="myTab" role="tablist">
  <li class="nav-item">
    <a class="nav-link active" id="home-tab" data-toggle="tab" href="#home"
    role="tab" aria-controls="home" aria-selected="true">Home</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" id="profile-tab" data-toggle="tab" href="#profile"
    role="tab" aria-controls="profile" aria-selected="false">Profile</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" id="contact-tab" data-toggle="tab" href="#contact"
    role="tab" aria-controls="contact" aria-selected="false">Contact</a>
  </li>

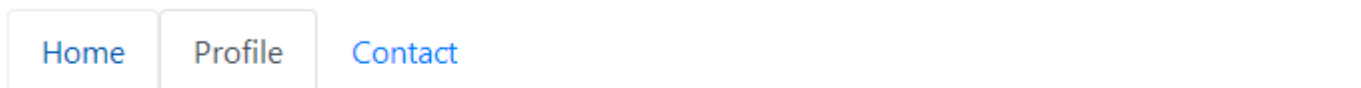
```

```

</ul>
<div class="tab-content" id="myTabContent">
  <div class="tab-pane fade show active" id="home" role="tabpanel"
aria-labelledby="home-tab">...</div>
  <div class="tab-pane fade" id="profile" role="tabpanel"
aria-labelledby="profile-tab">...</div>
  <div class="tab-pane fade" id="contact" role="tabpanel"
aria-labelledby="contact-tab">...</div>
</div>

```

如上所示，即可以使用标记，也可以基于 <nav>标签设计。上面案例是基于容器的选项卡（滑动门），下面案例是基于<nav>容器的选项卡（滑动门）：



Nulla est ullamco ut irure incididunt nulla Lorem Lorem minim irure officia enim reprehenderit. M
adipisicing exercitation ipsum. Nostrud ut anim non exercitation velit laboris fugiat cupidatat. Cor
ullamco magna consequat voluptate minim amet aliquip ipsum aute laboris nisi. Labore labore ve
magna in cupidatat dolore magna irure esse tempor ad mollit. Dolore commodo nulla minim ame
ullamco voluptate nisi commodo ea sit eu.

<

```

nav>
  <div class="nav nav-tabs" id="nav-tab" role="tablist">
    <a class="nav-item nav-link active" id="nav-home-tab" data-toggle="tab"
href="#nav-home" role="tab" aria-controls="nav-home"
aria-selected="true">Home</a>
    <a class="nav-item nav-link" id="nav-profile-tab" data-toggle="tab"
href="#nav-profile" role="tab" aria-controls="nav-profile"
aria-selected="false">Profile</a>
    <a class="nav-item nav-link" id="nav-contact-tab" data-toggle="tab"
href="#nav-contact" role="tab" aria-controls="nav-contact"
aria-selected="false">Contact</a>
  </div>
</nav>
<div class="tab-content" id="nav-tabContent">
  <div class="tab-pane fade show active" id="nav-home" role="tabpanel"
aria-labelledby="nav-home-tab">...</div>
  <div class="tab-pane fade" id="nav-profile" role="tabpanel"
aria-labelledby="nav-profile-tab">...</div>
  <div class="tab-pane fade" id="nav-contact" role="tabpanel"
aria-labelledby="nav-contact-tab">...</div>
</div>

```

选项卡（滑动门）组件也适用于胶囊式导航样式。

[Home](#)[Profile](#)[Contact](#)

Ad pariatur nostrud pariatur exercitation ipsum ipsum culpa mollit commodo mollit ex. Aute sunt incididunt amet deserunt pariatur do. Aliquip ex eiusmod voluptate exercitation cillum id incididunt elit sunt. Qui minim sit magna velit Lorem amet exercitation duis deserunt. Anim id labore elit adipisicing ut in id occaecat pariatur ut ullamco ea

```
<ul class="nav nav-pills mb-3" id="pills-tab" role="tablist">
  <li class="nav-item">
    <a class="nav-link active" id="pills-home-tab" data-toggle="pill"
href="#pills-home" role="tab" aria-controls="pills-home"
aria-selected="true">Home</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" id="pills-profile-tab" data-toggle="pill"
href="#pills-profile" role="tab" aria-controls="pills-profile"
aria-selected="false">Profile</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" id="pills-contact-tab" data-toggle="pill"
href="#pills-contact" role="tab" aria-controls="pills-contact"
aria-selected="false">Contact</a>
  </li>
</ul>
<div class="tab-content" id="pills-tabContent">
  <div class="tab-pane fade show active" id="pills-home" role="tabpanel"
aria-labelledby="pills-home-tab">...</div>
  <div class="tab-pane fade" id="pills-profile" role="tabpanel"
aria-labelledby="pills-profile-tab">...</div>
  <div class="tab-pane fade" id="pills-contact" role="tabpanel"
aria-labelledby="pills-contact-tab">...</div>
</div>
```

还可以做成垂直形式：

[Home](#)[Profile](#)[Messages](#)[Settings](#)

Culpa dolor voluptate do laboris laboris irure reprehenderit id incididunt duis pariatur mollit aute magna pariatur consectetur. Eu veniam duis non ut dolor deserunt commodo et minim in quis laboris ipsum velit id veniam. Quis ut consectetur adipisicing officia excepteur non sit. Ut et elit aliquip labore Lorem enim eu. Ullamco mollit occaecat dolore ipsum id officia mollit qui esse anim eiusmod do sint minim consectetur qui.

```
<div class="nav flex-column nav-pills" id="v-pills-tab" role="tablist"
aria-orientation="vertical">
  <a class="nav-link active" id="v-pills-home-tab" data-toggle="pill"
href="#v-pills-home" role="tab" aria-controls="v-pills-home"
aria-selected="true">Home</a>
  <a class="nav-link" id="v-pills-profile-tab" data-toggle="pill"
href="#v-pills-profile" role="tab" aria-controls="v-pills-profile"
aria-selected="false">Profile</a>
```

```

    <a class="nav-link" id="v-pills-messages-tab" data-toggle="pill"
href="#v-pills-messages" role="tab" aria-controls="v-pills-messages"
aria-selected="false">Messages</a>
    <a class="nav-link" id="v-pills-settings-tab" data-toggle="pill"
href="#v-pills-settings" role="tab" aria-controls="v-pills-settings"
aria-selected="false">Settings</a>
</div>
<div class="tab-content" id="v-pills-tabContent">
    <div class="tab-pane fade show active" id="v-pills-home" role="tabpanel"
aria-labelledby="v-pills-home-tab">...</div>
    <div class="tab-pane fade" id="v-pills-profile" role="tabpanel"
aria-labelledby="v-pills-profile-tab">...</div>
    <div class="tab-pane fade" id="v-pills-messages" role="tabpanel"
aria-labelledby="v-pills-messages-tab">...</div>
    <div class="tab-pane fade" id="v-pills-settings" role="tabpanel"
aria-labelledby="v-pills-settings-tab">...</div>
</div>

```

使用数据属性

只要在远素上指定 `data-toggle="tab"` 或 `data-toggle="pill"` 即可启动选项卡或胶囊式导航，而无需编写任何 JavaScript，并可在 `.nav-tabs` 或 `.nav-pills` 使用这些数据属性。

```

<!-- Nav tabs -->
<ul class="nav nav-tabs" id="myTab" role="tablist">
    <li class="nav-item">
        <a class="nav-link active" id="home-tab" data-toggle="tab" href="#home"
role="tab" aria-controls="home" aria-selected="true">Home</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" id="profile-tab" data-toggle="tab" href="#profile"
role="tab" aria-controls="profile" aria-selected="false">Profile</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" id="messages-tab" data-toggle="tab" href="#messages"
role="tab" aria-controls="messages" aria-selected="false">Messages</a>
    </li>
    <li class="nav-item">
        <a class="nav-link" id="settings-tab" data-toggle="tab" href="#settings"
role="tab" aria-controls="settings" aria-selected="false">Settings</a>
    </li>
</ul>

<!-- Tab panes -->
<div class="tab-content">
    <div class="tab-pane active" id="home" role="tabpanel"
aria-labelledby="home-tab">...</div>

```



```
<div class="tab-pane" id="profile" role="tabpanel"
aria-labelledby="profile-tab">...</div>
<div class="tab-pane" id="messages" role="tabpanel"
aria-labelledby="messages-tab">...</div>
<div class="tab-pane" id="settings" role="tabpanel"
aria-labelledby="settings-tab">...</div>
</div>
```

通过 JavaScript

通过 JavaScript 启用可选标签（每个选项卡需要单独激活）：

Copy

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

可以通过以下几种方式激活标签：

Copy

```
$('#myTab a[href="#profile"]').tab('show') // Select tab by name
$('#myTab li:first-child a').tab('show') // Select first tab
$('#myTab li:last-child a').tab('show') // Select last tab
$('#myTab li:nth-child(3) a').tab('show') // Select third tab
```

Fade 淡入淡出

要使标签淡入淡出，请添加 `.fade` 到每个 `.tab-pane`，第一个选项卡窗格还必须定义 `.show` 使默认初始内容可见。

Copy

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

方法

异步传输与转换

所有的 API 方法都支持**异步传输**，且一旦**转换开始**，在结束之前它们会返回给调用者，直到完成这个动画。另外，**过渡**组件上的方法将被忽略。

请参阅 [JavaScript 文档](#) 以了解更多信息。

`$.tab` 标签

激活选项卡元素和内容容器。标签应该具有 `data-target` 或 `href` 定位 DOM 中的容器节点。

Copy

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

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

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

`.tab('show')` 事件

选择给定的选项卡并显示其关联的窗格。之前选择的任何其他选项卡将被取消选中，并且其关联的窗格将被隐藏、在选项卡窗格实际显示之前（即在 `shown.bs.tab` 事件发生之前）返回到调用者。

Copy

```
$('#someTab').tab('show')
```

```
.tab('dispose')
```

D 选择给定的选项卡并显示其关联的窗格。之前选择的任何其他选项卡将被取消选中，并且其关联的窗格将被隐藏。在选项卡窗格实际显示之前（即在 `shown.bs.tab` 事件发生之前）返回到调用者。

事件

When showing a new tab, the events fire in the following order:

- 1. `hide.bs.tab` (当前被激活的分页标签上。
- 2. `show.bs.tab` 特显示的分页标签上)
- 3. `hidden.bs.tab` (在上一启动的标签选项卡上，与 `hide.bs.tab` 事件相同。
- 4. `shown.bs.tab` (在刚刚显示的标签先项卡上，与 `show.bs.tab` 事件相同。

如果没有选项卡标签已经启劝，那么 `hide.bs.tab` 和 `hidden.bs.tab` 事件不会被触发。

事件类型	描述
show.bs.tab	这个事件在一个标签选项卡内容显示上触发，但在选项卡显示之前，使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 来分别定位选项卡和上一个选项卡标签（如果可用）。
shown.bs.tab	这个事件在一个标签选项卡显示之后触发，使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 来分别定位选项卡和上一个启用的选项卡标签（如果可用）。
hide.bs.tab	在显示新标签选项卡之后触发此事件（因此以前的活动选项卡标签被隐藏），使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 来分别定位前一个使用的选项卡 and 新的即将启用的选项卡（如果有）。
hidden.bs.tab	在显示新标签选项卡之后触发此事件（因此以前的活动选项卡标签被隐藏），使用 <code>event.target</code> 和 <code>event.relatedTarget</code> 来分别定位前一个使用的选项卡 and 新启用的选项卡。

Copy

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

导航栏(navbar)

导航栏是一个将商标、导航以及别的元素简单放置到一个简洁导航页头的容器代码组合，它很容易扩展，而且在折叠板插件的帮助下，它可以轻松与其它内容整合。

运行原理

这些是你开始使用导航条之前需要知道的东西：

- 导航栏需要使用 `.navbar` 来定义, 并使用 `.navbar-expand{-sm|-md|-lg|-xl}` 用于响应式布局以及使用 [配色方案](#) Class 。
- 导航栏默认内容是流式的，使用 [containers 容喇](#)来限制它们的水平宽度。
- 使用我们提供的 [间隙间距](#) 和 [flex](#) 布局 classes 来定义导航栏中元素的间距和对齐。
- Navbars 导航栏默认支持响应式，在修改上也很容易，你可以使用轻松的来定义它们-这取决于我们提供的 JavaScript 插件。
- 打印时，导航栏默认隐藏。如果需要打印显示，可以加入 `.d-print` 样式到 `.navbar` 中，点此参阅 [display 块元素](#) 通用 class 定义。
- 使用 `<nav>` 导航通用元素来确保可访问性（易用性），或者如果使用更通用的元素，例如 `<div>` 添加一个 `role="navigation"`，可以为使用者的辅助浏览提供明确标识。

继续阅读以获取支持的子组件的示例和列表。

支持的内容

Navbar 导航栏内置支持少量子组件。根据需要从以下选择：

- `.navbar-brand` 为您的公司，产品或项目名称。
- `.navbar-nav` 提供完整的高和轻便的导航（包括对下拉菜单的支持）。
- `.navbar-toggler` 用於我們的折疊插件和其他 [navigation toggling](#) 行為。
- `.form-inline` 用于任何表单控件和操作。
- `.navbar-text` 用于添加垂直居中的文本字符串。
- `.collapse.navbar-collapse` 用于通过父断点进行分组和隐藏导航列内容。

以下是一个自动在 **lg**（大）断点处的自动响应轻型导航栏中包含的所有子组件的示例。

Navbar Home Link Dropdown ▾ Disabled

Search

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"
aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle" href="#" id="navbarDropdown"
role="button" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
          Dropdown
        </a>
        <div class="dropdown-menu" aria-labelledby="navbarDropdown">
          <a class="dropdown-item" href="#">Action</a>
          <a class="dropdown-item" href="#">Another action</a>
          <div class="dropdown-divider"></div>
          <a class="dropdown-item" href="#">Something else here</a>
        </div>
      </li>
      <li class="nav-item">
        <a class="nav-link disabled" href="#">Disabled</a>
      </li>
    </ul>
    <form class="form-inline my-2 my-lg-0">
      <input class="form-control mr-sm-2" type="search" placeholder="Search"
aria-label="Search">
```

```

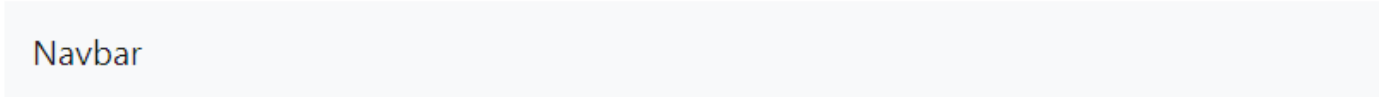
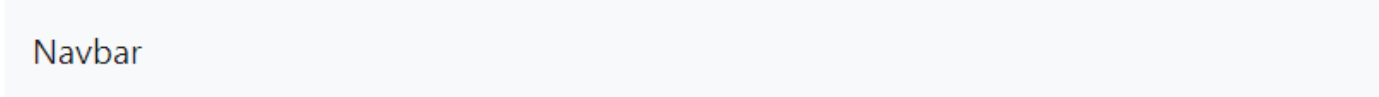
    <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
  </form>
</div>
</nav>

```

This example uses [color](#) (`bg-light`) and [spacing](#) (`my-2`, `my-lg-0`, `mr-sm-0`, `my-sm-0`) utility classes.

品牌

`.navbar-brand` 可以用于大多数元素，但对于链接最有效，因为某些元素可能需要通用样式类别 `class` 或自定义样式。



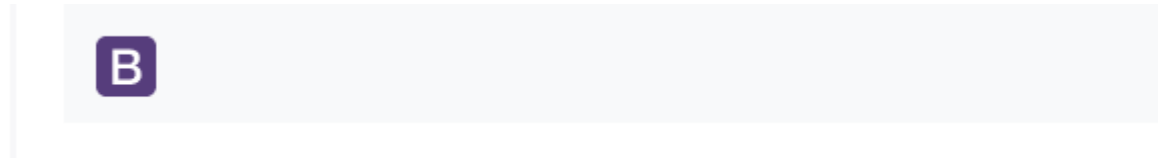
```

<!-- As a link -->
<nav class="navbar navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
</nav>

<!-- As a heading -->
<nav class="navbar navbar-light bg-light">
  <span class="navbar-brand mb-0 h1">Navbar</span>
</nav>

```

如果你喜欢，可以完全不使用列表法来做导航。

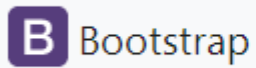


```

<!-- Just an image -->
<nav class="navbar navbar-light bg-light">
  <a class="navbar-brand" href="#">
    
  </a>
</nav>

```

[Bootstrap](#)



```
<!-- Image and text -->
<nav class="navbar navbar-light bg-light">
  <a class="navbar-brand" href="#">
    
    Bootstrap
  </a>
</nav>
```

nav 导航

导航栏链接建立在我们的 `.nav` 上，享有使用专属的 class 样式，并可以使用 [toggler 切换触发器](#) 来进行响应式切换，在导航栏中的元件，也装饰占用更多的水平空间，以保持导览列内容安全对齐。

活动状态指示：用 `.active` 表示当前页面，可直接应用于 `.nav-link` 或 `.nav-item` 上。

Navbar Home Features Pricing Disabled

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false"
aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarNav">
    <ul class="navbar-nav">
      <li class="nav-item active">
        <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Features</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Pricing</a>
```

```

    </li>
    <li class="nav-item">
      <a class="nav-link disabled" href="#">Disabled</a>
    </li>
  </ul>
</div>
</nav>

```

如果你喜欢（或需要），也可以不使用 ul、ol 式的列（直接用 A 其它元素作为列表子项-译者注）。

Navbar Home Features Pricing Disabled

```

<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup"
aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
    <div class="navbar-nav">
      <a class="nav-item nav-link active" href="#">Home <span
class="sr-only">(current)</span></a>
      <a class="nav-item nav-link" href="#">Features</a>
      <a class="nav-item nav-link" href="#">Pricing</a>
      <a class="nav-item nav-link disabled" href="#">Disabled</a>
    </div>
  </div>
</nav>

```

您还可以在导航栏中使用下拉列表，下拉菜单最好使用一个菜单进行位置定位包裹，请确保使用单独的元素嵌套 .nav-item 和 .nav-link，如下所示。

Navbar Home Features Pricing Dropdown link ▾

Action

```

<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>

```



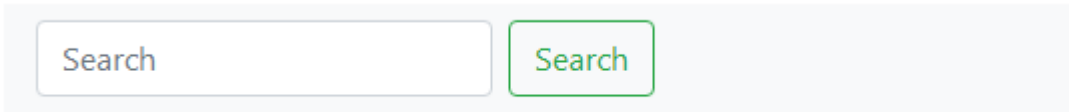
```

<button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarNavDropdown" aria-controls="navbarNavDropdown"
aria-expanded="false" aria-label="Toggle navigation">
  <span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarNavDropdown">
  <ul class="navbar-nav">
    <li class="nav-item active">
      <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Features</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#">Pricing</a>
    </li>
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" href="#" id="navbarDropdownMenuLink"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
        Dropdown link
      </a>
      <div class="dropdown-menu" aria-labelledby="navbarDropdownMenuLink">
        <a class="dropdown-item" href="#">Action</a>
        <a class="dropdown-item" href="#">Another action</a>
        <a class="dropdown-item" href="#">Something else here</a>
      </div>
    </li>
  </ul>
</div>
</nav>

```

Form 表单

在导航栏中使用 `.form-inline` 放置各种表单控制元件和组件。



The image shows a horizontal search bar. On the left is a text input field with the placeholder text "Search". To the right of the input field is a button with the text "Search". The entire search bar is contained within a light gray rectangular box.

```

<nav class="navbar navbar-light bg-light">

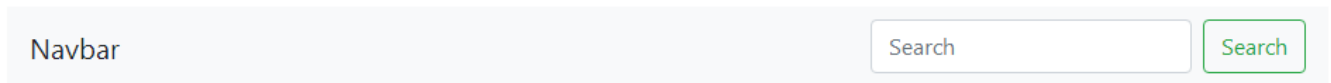
```

```

<form class="form-inline">
  <input class="form-control mr-sm-2" type="search" placeholder="Search"
aria-label="Search">
  <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
</form>
</nav>

```

根据需要将内联表单引用系统提供的内容与对齐等 class 样式。



```

<nav class="navbar navbar-light bg-light justify-content-between">
  <a class="navbar-brand">Navbar</a>
  <form class="form-inline">
    <input class="form-control mr-sm-2" type="search" placeholder="Search"
aria-label="Search">
    <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
  </form>
</nav>

```

还可以引用 `input-group` 输入框组控件：

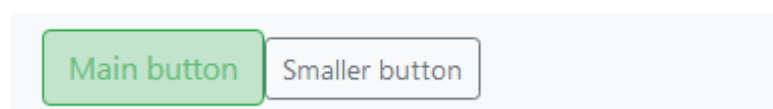


```

<nav class="navbar navbar-light bg-light">
  <form class="form-inline">
    <div class="input-group">
      <div class="input-group-prepend">
        <span class="input-group-text" id="basic-addon1">@</span>
      </div>
      <input type="text" class="form-control" placeholder="Username"
aria-label="Username" aria-describedby="basic-addon1">
    </div>
  </form>
</nav>

```

有的导航中需要用到各种按钮，可以使用通用样式 Class 来作居中对齐处理。



```

<nav class="navbar navbar-light bg-light">
  <form class="form-inline">
    <button class="btn btn-outline-success" type="button">Main button</button>
    <button class="btn btn-sm btn-outline-secondary" type="button">Smaller
button</button>
  </form>
</nav>

```

Text 文本处理

可以使用 `.navbar-text` 选择器来包裹文字-这已经对文本字符串的垂直对齐、水平间距作了处理优化。

Navbar text with an inline element

```

<nav class="navbar navbar-light bg-light">
  <span class="navbar-text">
    Navbar text with an inline element
  </span>
</nav>

```

根据需要与其它元件或通用样式定义组合使用。

Navbar w/ text Home Features Pricing

Navbar text with an inline element

```

<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar w/ text</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarText" aria-controls="navbarText" aria-expanded="false"
aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarText">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
        <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="#">Features</a>
      </li>
      <li class="nav-item">

```

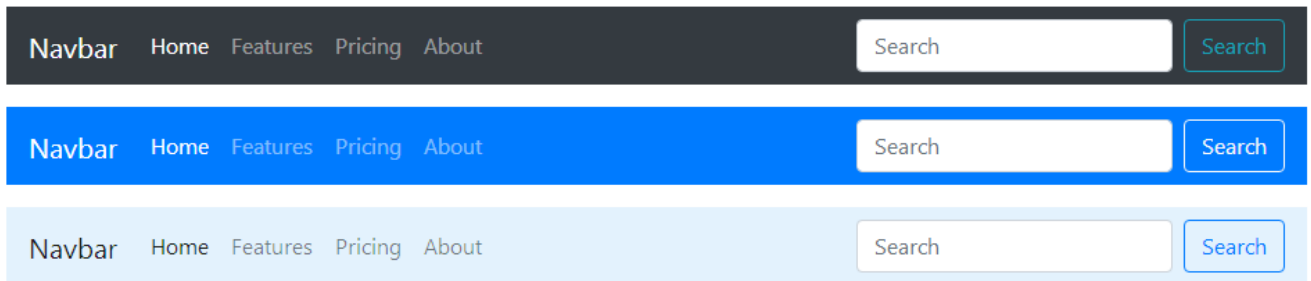
```

        <a class="nav-link" href="#">Pricing</a>
      </li>
    </ul>
    <span class="navbar-text">
      Navbar text with an inline element
    </span>
  </div>
</nav>

```

Color 颜色选择器(配色方案)

基于主题类 `class` 和 `background-color` 通用样式 `class` 定义，导航栏的配色方案和主题选择从未如此简单！你可以选择 `.navbar-light` 来定义导航颜色反转（强黑白对比），也可以使用 `.navbar-dark` 用于深色背景定义，然后再引用 `.bg-*` 类通用定义来进行大小处理。



```

<nav class="navbar navbar-dark bg-dark">
  <!-- Navbar content -->
</nav>

<nav class="navbar navbar-dark bg-primary">
  <!-- Navbar content -->
</nav>

<nav class="navbar navbar-light" style="background-color: #e3f2fd;">
  <!-- Navbar content -->
</nav>

```

.Container 主内容-容器

你可以把导航条包裹在一个 `.container` 容器中，从而使之在网页中呈现居中效果（或在导航栏内部居中）--虽然这不是强制的。

Navbar

```

<div class="container">
  <nav class="navbar navbar-expand-lg navbar-light bg-light">

```

```
<a class="navbar-brand" href="#">Navbar</a>
</nav>
</div>
```

When the container is within your navbar, its horizontal padding is removed at breakpoints lower than your specified `.navbar-expand{-sm|-md|-lg|-xl}` class. This ensures we're not doubling up on padding unnecessarily on lower viewports when your navbar is collapsed.

```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <div class="container">
    <a class="navbar-brand" href="#">Navbar</a>
  </div>
</nav>
```

定位

使用系统提供的 [position 位置间距定位通用样式](#) 可以使导航栏呈现出随浏览器滚动的效果（非固定位置），可选的流动可以包括固定在顶部、固定在底部、或粘到顶部（与页面滚动，直到顶部并停留到那里）。固定导航栏可以使用 `position: fixed` 属性，这意味着它们从 DOM 的正常流动和拉动可能需要自定义的 CSS(如在 `<body>` 上定义 `padding-top`)，以防止其重叠覆盖了其它元素。

注意：在 `.sticky-top` 使用 `position: sticky`, [目前不支持所有常用浏览器](#)。

默认

```
<nav class="navbar navbar-light bg-light">
  <a class="navbar-brand" href="#">默认</a>
</nav>
```

固定在顶部

```
<nav class="navbar fixed-top navbar-light bg-light">
  <a class="navbar-brand" href="#">固定在顶部</a>
</nav>
```

固定在底部

```
<nav class="navbar fixed-bottom navbar-light bg-light">
  <a class="navbar-brand" href="#">固定在底部</a>
</nav>
```

呈现粘性(随屏滚动)于浏览器窗口顶部

```
<nav class="navbar sticky-top navbar-light bg-light">
  <a class="navbar-brand" href="#">呈现粘性(随屏滚动)于浏览器窗口顶部</a>
</nav>
```

响应式行为处理

当内容在按钮后面折叠时，导航栏可以使

用 `.navbar-toggler`、`.navbar-collapse` 和 `.navbar-expand{-sm|-md|-lg|-xl}` 的 class 样式来更改，结合其它常用样式，你可以根据需要定义显示或隐藏特定元素。

對於不需要折疊的導覽列，在導覽列中加入 `.navbar-expand`。對於總是折疊的導覽列，不要加任何 `.navbar-expand` class。对于永不崩溃.navbar-expand 的导航栏，请在导航栏上添加该类。对于总是崩溃的导航栏，不要添加任何.navbar-expand 类。对于不需要折叠（隐藏）的导航栏，请在导航栏上增加 `.navbar-expand` class 样式来定义，对于总是要折叠（隐藏）的导航栏，请不要加任何的 `.navbar-expand` class 样式。

Toggler 切换触发器

Navbar 下的 Toggler 是切换触发器（即手机模式下的 MENU 下拉项），它们默认是左对齐的。如果在它们中间定义一个同级的兄弟元素.navbar-brand，它们会自动对齐到窗口右边，反转你的品牌（LOGO 或主标题）元素的位置，以下是不同切换形式（位置）的示例（需要在移动窄屏下才能看到效果）：

没有在最下最小浏览界面中定义 `.navbar-brand` 的样式（MENU 切换按钮在默认左边）：

Hidden brand Home Link Disabled

Search

Search

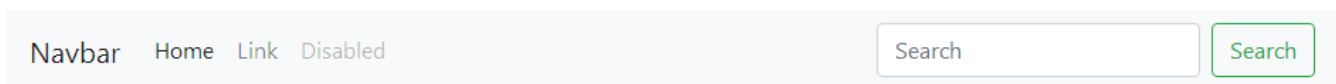
```
<nav class="navbar navbar-expand-lg navbar-light bg-light">
  <button class="navbar-toggler" type="button" data-toggle="collapse"
    data-target="#navbarTogglerDemo01" aria-controls="navbarTogglerDemo01"
    aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
    <a class="navbar-brand" href="#">Hidden brand</a>
    <ul class="navbar-nav mr-auto mt-2 mt-lg-0">
      <li class="nav-item active">
```

```

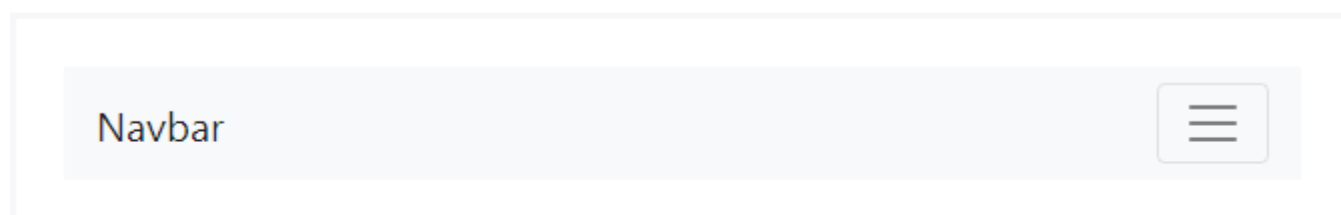
        <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
        <a class="nav-link disabled" href="#">Disabled</a>
    </li>
</ul>
<form class="form-inline my-2 my-lg-0">
    <input class="form-control mr-sm-2" type="search" placeholder="Search"
aria-label="Search">
    <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
</form>
</div>
</nav>

```

左侧有一个 LOGO(主标题)，右边是 MENU 切换按钮：



左侧有一个LOGO(主标题)，右边是MENU切换按钮：



```

<nav class="navbar navbar-expand-lg navbar-light bg-light">
    <a class="navbar-brand" href="#">Navbar</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarTogglerDemo02" aria-controls="navbarTogglerDemo02"
aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>

    <div class="collapse navbar-collapse" id="navbarTogglerDemo02">
        <ul class="navbar-nav mr-auto mt-2 mt-lg-0">
            <li class="nav-item active">

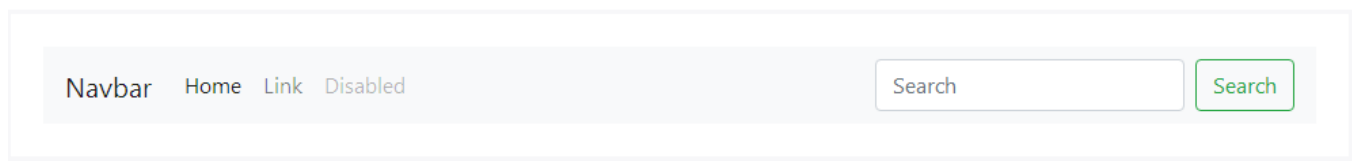
```

```

        <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
    </li>
    <li class="nav-item">
        <a class="nav-link" href="#">Link</a>
    </li>
    <li class="nav-item">
        <a class="nav-link disabled" href="#">Disabled</a>
    </li>
</ul>
<form class="form-inline my-2 my-lg-0">
    <input class="form-control mr-sm-2" type="search" placeholder="Search">
    <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
</form>
</div>
</nav>

```

右侧有一个 LOGO(主标题)，左边是 MENU 切换按钮：



```

<nav class="navbar navbar-expand-lg navbar-light bg-light">
    <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarTogglerDemo03" aria-controls="navbarTogglerDemo03"
aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>
    <a class="navbar-brand" href="#">Navbar</a>

    <div class="collapse navbar-collapse" id="navbarTogglerDemo03">
        <ul class="navbar-nav mr-auto mt-2 mt-lg-0">
            <li class="nav-item active">
                <a class="nav-link" href="#">Home <span
class="sr-only">(current)</span></a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="#">Link</a>
            </li>
            <li class="nav-item">
                <a class="nav-link disabled" href="#">Disabled</a>
            </li>
        </ul>
    </div>
</nav>

```



```

        </li>
    </ul>
    <form class="form-inline my-2 my-lg-0">
        <input class="form-control mr-sm-2" type="search" placeholder="Search"
aria-label="Search">
        <button class="btn btn-outline-success my-2 my-sm-0"
type="submit">Search</button>
    </form>
</div>
</nav>

```

扩展导航区内容

使用 `id` 和 `data-target` 搭配，很容易达成：



```

<div class="pos-f-t">
    <div class="collapse" id="navbarToggleExternalContent">
        <div class="bg-dark p-4">
            <h4 class="text-white">Collapsed content</h4>
            <span class="text-muted">Toggleable via the navbar brand.</span>
        </div>
    </div>
    <nav class="navbar navbar-dark bg-dark">
        <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarToggleExternalContent"
aria-controls="navbarToggleExternalContent" aria-expanded="false"
aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
    </nav>
</div>

```

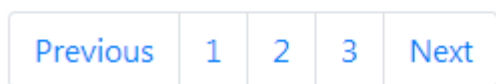
分页(Pagination)

用于指导定义分页以及显示样式定义的一些示例和文档。

概览

我们使用大规格 block 块的 A 链接进行分页样式呈现，使链接难以忽略、易于扩展，且提供更大的点击区域使用户易于上手。分页是使用 list 列表元素构建的，因此屏幕阅读器可以读出链接的数量，使用 `<nav>` 元素将其识别为屏幕阅读器和其它辅助技术的导航组件并提供辅助支持。

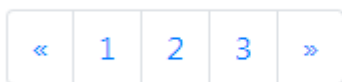
此外，网页可能有不止一个这样的导航部分(存在多个导航组件)，建议您为 `<nav>` 提供一个 `aria-label` 的描述，以反映其功能，例如：如果分页组件用于一组搜索结果导览，则标签可以是：
`aria-label="Search results pages"`。



```
<nav aria-label="Page navigation example">
  <ul class="pagination">
    <li class="page-item"><a class="page-link" href="#">Previous</a></li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item"><a class="page-link" href="#">2</a></li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item"><a class="page-link" href="#">Next</a></li>
  </ul>
</nav>
```

使用图标

寻找使用图标或符号代替文本的一些分页链接？一定要提供适当的屏幕阅读器支持 `aria` 属性和 `.sr-only` 实用程序。想要使用图示或符号代替某些分页連結的文字？使用 `aria` 屬性和 `.sr-only` 通用類別提供螢幕閱讀器的支援。想使用图标或符合代替分页链接中的文字（如首页、下一页），如下图设置就可以（使用 `aria` 属性和 `.sr-only` 通用样式来为屏幕阅读器提供友好访问支持。



```
<nav aria-label="Page navigation example">
  <ul class="pagination">
    <li class="page-item">
```

```

<a class="page-link" href="#" aria-label="Previous">
  <span aria-hidden="true">&laquo;</span>
  <span class="sr-only">Previous</span>
</a>
</li>
<li class="page-item"><a class="page-link" href="#">1</a></li>
<li class="page-item"><a class="page-link" href="#">2</a></li>
<li class="page-item"><a class="page-link" href="#">3</a></li>
<li class="page-item">
  <a class="page-link" href="#" aria-label="Next">
    <span aria-hidden="true">&raquo;</span>
    <span class="sr-only">Next</span>
  </a>
</li>
</ul>
</nav>

```

禁用和活动状态定义

分页组件可以根据不同的情况进行定制，使用 `code class="highlighter-rouge">.disabled` 显示为分页链接不可用（禁用无效），使用 `.active` 可指引标示当前所在的分页。

`.disabled` 使用 `pointer-events: none` 来禁用 `<a>` 的链接功能，但该 CSS 属性尚未标准化（使用的时候要注意浏览器兼容性调式-译者注）。

此外，即使在支持 `pointer-events: none` 的浏览器中，键盘导航仍然不受影响，这意味着键盘使用者和辅助技术浏览器用户仍然可以启动这些被“禁用”的链接。为了安全起见，请在这些链接上添加一个 `tabindex="-1"` 定义并使用自定义 JavaScript 来完全禁用其功能。



```

<nav aria-label="...">
  <ul class="pagination">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1">Previous</a>
    </li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item active">
      <a class="page-link" href="#">2 <span class="sr-only">(current)</span></a>
    </li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
      <a class="page-link" href="#">Next</a>
    </li>
  </ul>
</nav>

```

</nav>

您可以用 `` 替换有效或禁用的链接，或者在上一页/下一页箭头替代（省略）A 链接以删除点击功能，以保留原样式的同事防止键盘 focus 强游活。



```
<nav aria-label="...">
  <ul class="pagination">
    <li class="page-item disabled">
      <span class="page-link">Previous</span>
    </li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item active">
      <span class="page-link">
        2
        <span class="sr-only">(current)</span>
      </span>
    </li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
      <a class="page-link" href="#">Next</a>
    </li>
  </ul>
</nav>
```

规格尺寸

要更大或更小的分页，添加 `.pagination-lg` 或 `.pagination-sm` 样式定义可以获得大规格或小规格尺寸的分页组件。

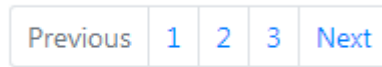


```
<nav aria-label="...">
  <ul class="pagination pagination-lg">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1">Previous</a>
    </li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item"><a class="page-link" href="#">2</a></li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
```

```

    <a class="page-link" href="#">Next</a>
  </li>
</ul>
</nav>

```



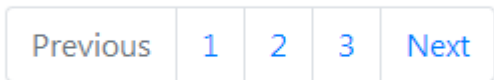
```

<nav aria-label="...">
  <ul class="pagination pagination-sm">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1">Previous</a>
    </li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item"><a class="page-link" href="#">2</a></li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
      <a class="page-link" href="#">Next</a>
    </li>
  </ul>
</nav>

```

对齐

使用 [flexbox 弹性布局通用样式](#)，可更改分页组件的对齐方式。



```

<nav aria-label="Page navigation example">
  <ul class="pagination justify-content-center">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1">Previous</a>
    </li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item"><a class="page-link" href="#">2</a></li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
      <a class="page-link" href="#">Next</a>
    </li>
  </ul>

```

```
</ul>
</nav>
```

Previous	1	2	3	Next
----------	---	---	---	------

```
<nav aria-label="Page navigation example">
  <ul class="pagination justify-content-end">
    <li class="page-item disabled">
      <a class="page-link" href="#" tabindex="-1">Previous</a>
    </li>
    <li class="page-item"><a class="page-link" href="#">1</a></li>
    <li class="page-item"><a class="page-link" href="#">2</a></li>
    <li class="page-item"><a class="page-link" href="#">3</a></li>
    <li class="page-item">
      <a class="page-link" href="#">Next</a>
    </li>
  </ul>
</nav>
```

POP 提示 (Popover)

将 Bootstrap 插件（如同 iOS 的渐变模态提示）添加到网站上的任何元素上，这是演示实例和设计文档。

概览

使用 popover 提示框插件时需要注意的事项：

- popover 提示框组件依赖 [Popper.js](#) 进行定位，在必须引入 [popper.min.js](#) 在 bootstrap.js 脚本之前，或者使用 [bootstrap.bundle.min.js](#) / [bootstrap.bundle.js](#)（这两个脚本中已经包含了 [Popper.js](#) 可以直接运行弹出提示框）。
- popover 提示框组件依赖于 [tooltips 提示冒泡插件](#) 提供状态提示框。
- 如果你要自行编译 JS，请 [包括 util.js](#) 脚本文件。
- 由于性能原因，popover 提示框组件是可选的，所以 **你必须自己将它们初始化**才能启动生效。
- 零长度的 **title** 和 **content** 不会显示为一个弹出提示框。
- 指定 **container: 'body'** 定义，以避免在更复杂的组件（如输入框组、按钮组等）中产生呈现问题。
- 在隐藏元素上触发弹出提示框是无效（不起作用）的。
- **.disabled** 或 **disabled** 元素的弹出提示框触发点击按钮，必须在在外层父元素上。
- 如果从一个跨多行的链接上触发提示冒泡，提示冒泡将居中。在 `<a>` 上使用 **white-space: nowrap;** 可以避免这种情况。
- 必须先隐藏弹出提示框，然后才能从 DOM 中删除相应的元素（以完成弹出关闭完整行为）。

都记住了？非常棒，下面让我们通过一系列的实例来了解其运作机理。

示例：在任意位置启用弹出窗口

One way to initialize all popovers on a page would be to select them by their **data-toggle** attribute:

Copy

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

示例：使用 **container** 容器选项

当你在一个父元素上有一些样式与提示框产生样式干扰，你可以指定一个自定义的 **container** 容器，这样提示框的 HTML 将出现在这个元素内部了。

Copy

```
$(function () {  
  $('.example-popover').popover({  
    container: 'body'  
  })  
})
```

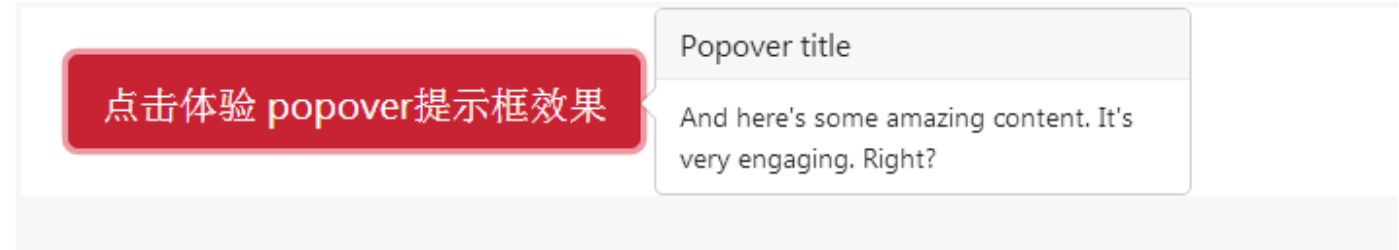
静态提示框

可以使用四个选项：顶部、右部、底部、左部对齐（指向）。



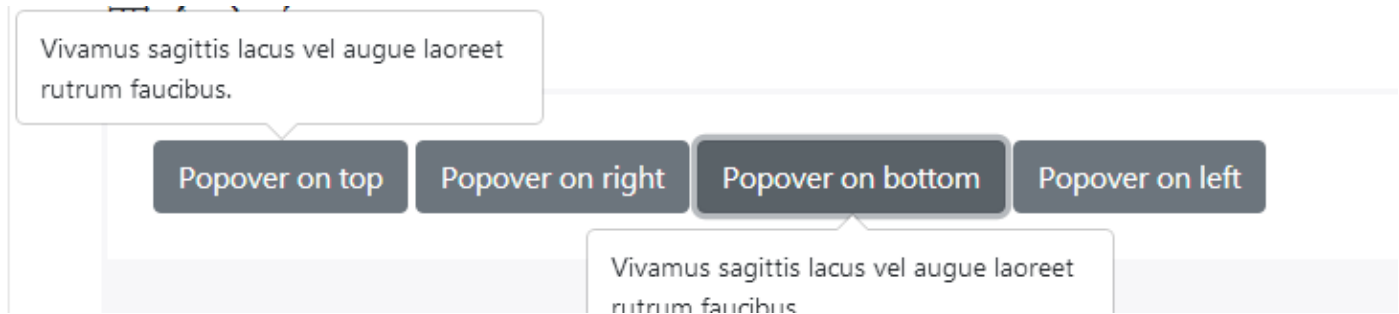
现场演示

点击体验 popover 提示框效果



```
<button type="button" class="btn btn-lg btn-danger" data-toggle="popover" title="Popover title" data-content="And here's some amazing content. It's very engaging. Right?">点击体验 popover 提示框效果</button>
```

四个方向



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



```
    Popover on top
</button>
```

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

```
    Popover on right
</button>
```

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

```
    Popover on bottom

</button>
```

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

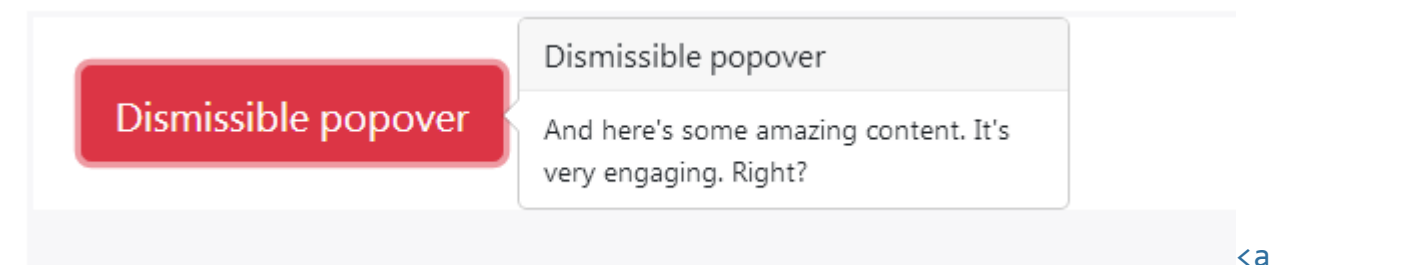
```
    Popover on left
</button>
```

在下次点击时收回

使用 `focus` 触发器，达到提示必须在用户下一次点击时才能收回（移除）提示事件。

下一步点击所需要的标记

为正确处理（兼容）跨浏览器和跨平台行为，你必须使用则必须使用 `<a>` 标签，而 不是标签，你还必须包括一个 `tabindex` 属性。



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

```
Copy
$( '.popover-dismiss' ).popover({
  trigger: 'focus'
})
```

禁用的元素

具有 `disabled` 属性的元素不是交互式的这意味着用户不能悬停或点击它们来触发弹出窗口（或工具提示）。 作为一种解决方法，您需要从包装器 `<div>` or `` 中触发弹出窗口，并覆盖 `disabled` 元素上的指针事件。

对于禁用的弹出式触发器，您也可能更喜欢 `data-trigger="hover"`，以便弹出窗口显示为用户的直接视觉反馈，因为他们可能不希望单击禁用的元素。



```
<span class="d-inline-block" data-toggle="popover" data-content="Disabled popover">
  <button class="btn btn-primary" style="pointer-events: none;" type="button" disabled>Disabled button</button>
</span>
```

用法

利用 JavaScript 启动提示泡:

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

选项

可通过数据属性或 JavaScript 传递选项参数。对于数据属性，请将选项名称附加到 `data-` 上，如 `data-animation=""`。

名称	Type 类型	默认值	描述
animation	boolean	true	将 CSS 淡入淡出转换应用于 popover 提示框。
container	string element false	false	向一个特定元素追加提示框，如 <code>container: 'body'</code> 。这个选项候特别有用，因为它允许你将弹出提示定位在触发元素附近的内容中--这将防止在窗口调整大小的时候，提示框飘到远离（撑出）触发元素的位置。
content	string element function	""	如果 <code>data-content</code> 属性不存在，提供默认的 content 值。 如果提供了一个函数，调用这个函数时，函数的 <code>this</code> 引用被设置为附加提示框的元素。
delay	number object	0	延迟显示和隐藏弹出（ms） - 不适用于手动触发类型。 如果向这个选项提供一个数字值，显示和隐藏都会应用这个延迟。 Object 对象结构是: <code>delay: { "show": 500, "hide": 100 }</code>
html	boolean	false	向提示框插入 HTML。如果值为 false，将使用 jQuery

名称	Type 类型	默认值	描述
			的 text 方法向 DOM 插入内容。如果你担心跨站脚本攻击的话，请使用 text 。
placement	string function	'right'	<p>如何定位提示框：auto top bottom left right。如果指定了"auto"，它会动态地调整提示框的位置。举个例子，如果 placement 是"auto left"，提示框将尽可能地显示在左侧，否则显示在右侧。</p> <p>如果用一个函数来决定 placement，会调用提示框的 DOM 节点作为第一个参数，然后触发元素的 DOM 节点是第二个参数。this 将被设置为提示框的实例。</p>
selector	string false	false	如果提供了一个选择器，提示框将被委派给指定的目标。在实践中，它让动态添加的 HTML 内容也能附加提示框。请参阅 这里文档 以及 一个翔实的示例 。
template	string	'<div class="popover" role="tooltip"><div class="arrow"></div><h3 class="popover-header"></h3><div class="popover-body"></div></div>'	<p>在创建提示框时使用基本 HTML。</p> <p>动态提示框的 title 值将被注入到 .popover-header 中。</p> <p>动态提示框的 content 值将被注入到 .popover-body 中。</p> <p>.arrow 将变成提示框的箭头。</p> <p>最外层的包裹元素必须拥有 .popover class 样式定义</p>
title	string element function	''	<p>如果 title 属性不存在，提供默认 title 值。</p> <p>如果提供了一个函数，调用这个函数时，函数的 this 引用被设置为附加提示框的元素。</p>
trigger	string	'click'	如何触发提示框 - click hover focus manual。你可以传递多个触发器，用空格隔开它们。但是`manual`不能与别的触发器结合使用。
offset	number string	0	提示框相对于目标的偏移，欲知更多请查询 Tether 的偏移文档 。
fallbackPlacement	string array	'flip'	指定动态提示框在回调时使用哪个位置，有关更多信息请参阅 Popper.js 的 行为属性文档 。

单个 popovers 的数据属性

如上所述，可以通过使用数据属性来指定各个提示框的选项。

方法

异步传输与转换

所有的 API 方法都是**异步**的，并开始一个转换。一旦转换开始但在结束之前，它们就返回给调用者。另外，一个转换组件的方法调用将被忽略。

请参阅我们的 [JavaScript 文档](#) 以获取更多信息。

`$().popover(options)`

弹出提示框初始化。

`.popover('show')`

显示一个元素的提示框，在弹出**实际显示之前返回给调用者**（即在 `shown.bs.popover` 事件发生前），这被认为是弹出提示框的**手动触发**，标题和内容在弹出提示框不显示。

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

隐藏元素的提示框，在弹出提示框实际被隐藏之前返回给调用者（即在 `hidden.bs.popover` 事件发生前），这被认为是弹出提示框的**手动触发**。

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

切换元素的提示框，在 `popover` 提示框实际显示或隐藏之前返回给调用者，即在 `shown.bs.popover` 或 `hidden.bs.popover` 事件发生前，这被认为是弹出提示框的**手动触发**。

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

隐藏和销毁一个元素的提示框，使用委托授权（使用 [the selector option](#) 创建）的提示框不能在后代触发元素不被单独销毁。

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

默认情况下是启用的。

```
$('#element').popover('enable')  
.popover('disable')
```

删除要显示元素的提示框，只有在重新启用后，才能显示出提示框。

```
$('#element').popover('disable')  
.popover('toggleEnabled')
```

切换提示窗口的显示或隐藏功能。

```
$('#element').popover('toggleEnabled')  
.popover('update')
```

更新元素的提示框的位置。

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

Events 事件

事件	描述
show.bs.popover	当调用 <code>show</code> 实例方法时，此事件立即触发。
shown.bs.popover	当提示框显示时，会触发此事件（待 CSS 转换过渡完成）。
hide.bs.popover	当调用 <code>hide</code> 实例方法时，此事件立即触发。
hidden.bs.popover	当提示框隐藏后，会触发此事件（待 CSS 转换过渡完成）。
inserted.bs.popover	当提示框对用户来说最终完成隐藏时（需要等待 CSS 转换过渡完成），会触发该事件。当事件弹出模板被添加时， <code>show.bs.popover</code> 事件被触发。

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

进度条(Progress)

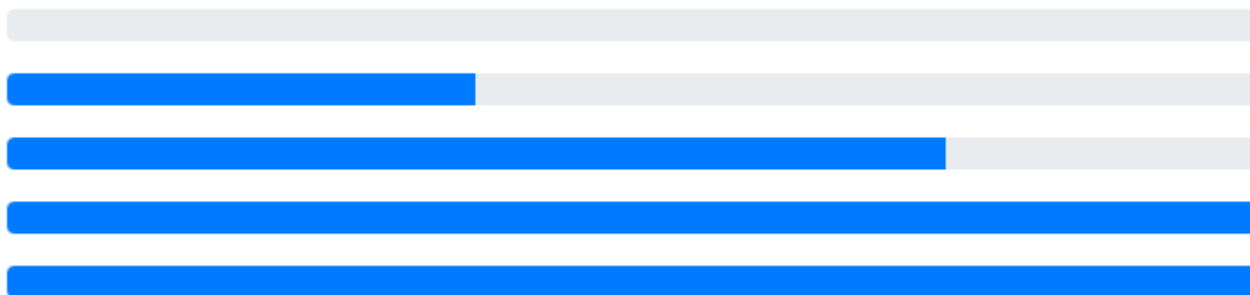
使用 Bootstrap 自定义进度条的文档和示例，支持平行条状堆叠、动画背景和文本标签。

运行原理

Progress 进度条组件由两个 HTML 元素、一些 CSS 宽度设置以及一些属性构建。我们不使用 [HTML5 <progress> 元素](#)，确保用户可以方便的平行堆叠进度条、加上动态效果，并在其上放置文本标签。

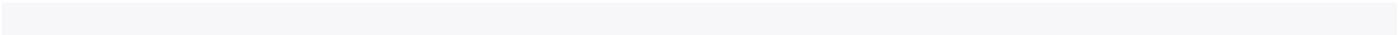
- 使用 `.progress` 容器来指定进度条的最大值。
- 使用 `.progress-bar` 来表示当前的进度。
- `.progress-bar` 要求的内嵌样式，使用全局实用 CSS 或自定义 CSS 来设置其宽度。
- `.progress-bar` 还需要一些 `role` and 与属性，使其访问友好（无障碍）。

把它们组合起来，可得到以下实例：



```
<div class="progress">
  <div class="progress-bar" role="progressbar" aria-valuenow="0"
aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 25%"
aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 50%"
aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 75%"
aria-valuenow="75" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 100%"
aria-valuenow="100" aria-valuemin="0" aria-valuemax="100"></div>
</div>
```

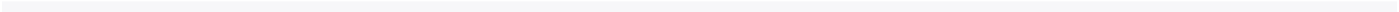
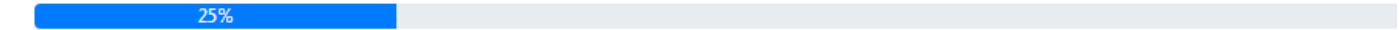
Bootstrap 提供了一些用于 [设置宽度的通用样式设置](#)。根据您的需要，这可能有助于快速配置进度条组件。



```
<div class="progress">
  <div class="progress-bar w-75" role="progressbar" aria-valuenow="75"
aria-valuemin="0" aria-valuemax="100"></div>
</div>
```

标签

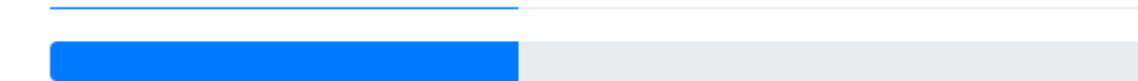
在 `.progress-bar` 中放置文字内容,可将标签添加到进度条中。



```
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 25%;"
aria-valuenow="25" aria-valuemin="0" aria-valuemax="100">25%</div>
</div>
```

高度

我们只在 `.progress` 上设置了一个 `height` 值,所以如果你改变了这个值,那么内部的 `.progress-bar` 高度会自动调整大小。



```
<div class="progress" style="height: 1px;">
  <div class="progress-bar" role="progressbar" style="width: 25%;"
aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress" style="height: 20px;">
  <div class="progress-bar" role="progressbar" style="width: 25%;"
aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
</div>
```

背景

使用背景通用样式 `Class` 样式来定义进度条的外观。



```
<div class="progress">
  <div class="progress-bar bg-success" role="progressbar" style="width: 25%"
aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar bg-info" role="progressbar" style="width: 50%"
aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar bg-warning" role="progressbar" style="width: 75%"
aria-valuenow="75" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar bg-danger" role="progressbar" style="width: 100%"
aria-valuenow="100" aria-valuemin="0" aria-valuemax="100"></div>
</div>
```

多进度条进度（嵌套）

如果有需要，可在进度组件中包含（嵌套）多个进度条。



```
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 15%"
aria-valuenow="15" aria-valuemin="0" aria-valuemax="100"></div>
  <div class="progress-bar bg-success" role="progressbar" style="width: 30%"
aria-valuenow="30" aria-valuemin="0" aria-valuemax="100"></div>
  <div class="progress-bar bg-info" role="progressbar" style="width: 20%"
aria-valuenow="20" aria-valuemin="0" aria-valuemax="100"></div>
</div>
```

条纹进度指示

将 `.progress-bar-striped` 添加到 `.progress-bar` 上，可以使用 CSS 渐变对背景颜色加上条纹效果。



```
<div class="progress">
  <div class="progress-bar progress-bar-striped" role="progressbar" style="width:
10%" aria-valuenow="10" aria-valuemin="0" aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-striped bg-success" role="progressbar"
style="width: 25%" aria-valuenow="25" aria-valuemin="0"
aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-striped bg-info" role="progressbar"
style="width: 50%" aria-valuenow="50" aria-valuemin="0"
aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-striped bg-warning" role="progressbar"
style="width: 75%" aria-valuenow="75" aria-valuemin="0"
aria-valuemax="100"></div>
</div>
<div class="progress">
  <div class="progress-bar progress-bar-striped bg-danger" role="progressbar"
style="width: 100%" aria-valuenow="100" aria-valuemin="0"
aria-valuemax="100"></div>
</div>
```

动画条纹进度指示

条纹渐变也可以做成动画效果，将 `.progress-bar-animated` 加到 `.progress-bar` 上，即实现 CSS3 绘制的从右到左的动画效果。

动画条纹进度条不适用于 **Opera 12 浏览器**——因为它不支持 CSS3 动画。



```
<div class="progress">
  <div class="progress-bar progress-bar-striped progress-bar-animated"
role="progressbar" aria-valuenow="75" aria-valuemin="0" aria-valuemax="100"
style="width: 75%"></div></div>
```

滚动监听 (Scrollspy)

滚动监听插件会根据滚动的位置，自动更新导航条的目标，以指示当前窗口中处于活动的状态（滚动在导航条下面的区域，查看 `active` 类的改变。弹出菜单的子项也同样会被高亮）。

运行原理

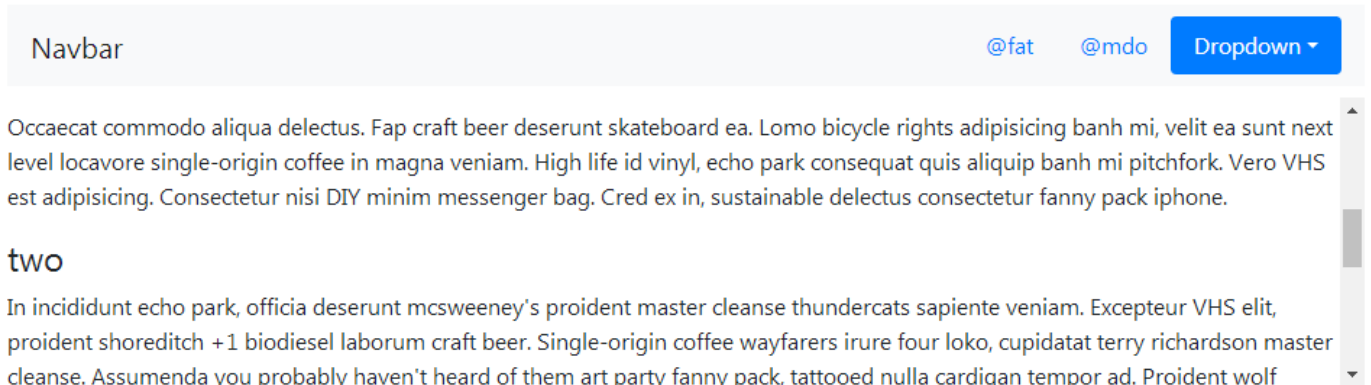
Scrollspy 滑动监控组件正常运行的几个要点:

- 如果从源代码构建 JS，请 [包括 util.js](#)。
- 它必须在 Bootstrap [nav 导航组](#) 或 [list group 列表组](#) 上使用。
- Scrollspy 需要在你监控的元素上使用 `position: relative;`，通常是 `<body>`。
- 当需要对 `<body>` 以外的元素进行监控时，请确保具有 `height` 高度值和 `overflow-y: scroll;` 属性。
- 锚点 (`<a>`) 是必须的，并且必须指向一个 `id` 上。

实施成功后，你的导航或列表群组将相应地更新，根据 `.active` 关联的目标，从一个项目移到另一个项目。

在 navbar 导航中的示例

滚动导航栏下方的区域，并观看活动列表的变化，下拉项目也会突出显示，如下例所示：



```
<nav id="navbar-example2" class="navbar navbar-light bg-light">
  <a class="navbar-brand" href="#">Navbar</a>
  <ul class="nav nav-pills">
    <li class="nav-item">
      <a class="nav-link" href="#fat">@fat</a>
    </li>
    <li class="nav-item">
      <a class="nav-link" href="#mdo">@mdo</a>
    </li>
    <li class="nav-item dropdown">
      <a class="nav-link dropdown-toggle" data-toggle="dropdown" href="#"
        role="button" aria-haspopup="true" aria-expanded="false">Dropdown</a>
      <div class="dropdown-menu">
        <a class="dropdown-item" href="#one">one</a>
```

```

        <a class="dropdown-item" href="#two">two</a>
        <div role="separator" class="dropdown-divider"></div>
        <a class="dropdown-item" href="#three">three</a>
    </div>
</li>
</ul>
</nav>
<div data-spy="scroll" data-target="#navbar-example2" data-offset="0">
    <h4 id="fat">@fat</h4>
    <p>...</p>
    <h4 id="mdo">@mdo</h4>
    <p>...</p>
    <h4 id="one">one</h4>
    <p>...</p>
    <h4 id="two">two</h4>
    <p>...</p>
    <h4 id="three">three</h4>
    <p>...</p>
</div>

```

嵌套的导航示例

Scrollspy 滚动监控也适用于嵌套的 `.nav`。如果是一个嵌套，`.nav` 是的 `.active` 状态，其父母也将是 `.active` 状态。滚动导航栏旁边的区域，并观看活动班级更改，如下例：

<div style="background-color: #f8f9fa; padding: 10px; border-radius: 5px;"> <div style="text-align: center; font-weight: bold; margin-bottom: 10px;">Navbar</div> <div style="background-color: #007bff; color: white; padding: 5px; text-align: center; margin-bottom: 5px;">Item 1</div> <div style="padding-left: 20px; color: #007bff;">Item 1-1</div> <div style="background-color: #007bff; color: white; padding: 5px; text-align: center; margin-bottom: 5px;">Item 1-2</div> <div style="padding-left: 20px; color: #007bff;">Item 2</div> <div style="padding-left: 20px; color: #007bff;">Item 3</div> <div style="padding-left: 40px; color: #007bff;">Item 3-1</div> <div style="padding-left: 40px; color: #007bff;">Item 3-2</div> </div>	<p> eiusmod dolor et eiusmod. Anim occaecat nulla in non consequat eiusmod velit incididunt.</p> <h3>Item 2</h3> <p>Quis magna Lorem anim amet ipsum do mollit sit cillum voluptate ex nulla tempor Laborum consequat non elit enim exercitation cillum aliqua consequat id aliqua. Es: ex consectetur mollit voluptate est in duis laboris ad sit ipsum anim Lorem. Incididunt veniam velit elit elit veniam Lorem aliqua quis ullamco deserunt sit enim elit aliqua esse irure. Laborum nisi sit est tempor laborum mollit labore officia laborum excepteur commodo non commodo dolor excepteur commodo. Ipsum fugiat ex est consectetur ipsum commodo tempor sunt in proident.</p> <h3>Item 3</h3> <p>Quis anim sit do amet fugiat dolor velit sit ea ea do reprehenderit culpa duis. Nostrum aliqua iusum fuaiat minim proident occaecat excepteur aliquip culpa aute tempor</p>
--	--

```

<nav id="navbar-example3" class="navbar navbar-light bg-light">
    <a class="navbar-brand" href="#">Navbar</a>
    <nav class="nav nav-pills flex-column">
        <a class="nav-link" href="#item-1">Item 1</a>
        <nav class="nav nav-pills flex-column">
            <a class="nav-link ml-3 my-1" href="#item-1-1">Item 1-1</a>
            <a class="nav-link ml-3 my-1" href="#item-1-2">Item 1-2</a>
        </nav>
    </nav>

```

```

<a class="nav-link" href="#item-2">Item2</a>
<a class="nav-link" href="#item-3">Item3</a>
<nav class="nav nav-pills flex-column">
  <a class="nav-link ml-3 my-1" href="#item-3-1">Item 3-1</a>
  <a class="nav-link ml-3 my-1" href="#item-3-2">Item 3-2</a>
</nav>
</nav>
</nav>

<div data-spy="scroll" data-target="#navbar-example3" data-offset="0">
  <h4 id="item-1">Item 1</h4>
  <p>...</p>
  <h5 id="item-1-1">Item 1-1</h5>
  <p>...</p>
  <h5 id="item-1-2">Item 2-2</h5>
  <p>...</p>
  <h4 id="item-2">Item 2</h4>
  <p>...</p>
  <h4 id="item-3">Item 3</h4>
  <p>...</p>
  <h5 id="item-3-1">Item 3-1</h5>
  <p>...</p>
  <h5 id="item-3-2">Item 3-2</h5>
  <p>...</p>
</div>

```

列表组示例

Scrollspy 滚动监听也适用于 `.list-group` 列表组，请滚动列表组旁边的区域，观看活动列表的变更：

Item 1

Item2

Item 3

Item 4

ex consectetur mollit voluptate est in duis laboris ad sit ipsum anim Lorem. Incidunt veniam velit elit elit veniam Lorem aliqua quis ullamco deserunt sit enim elit aliqua esse irure. Laborum nisi sit est tempor laborum mollit labore officia laborum excepteur commodo non commodo dolor excepteur commodo. Ipsum fugiat ex est consectetur ipsum commodo tempor sunt in proident.

Item 3

Quis anim sit do amet fugiat dolor velit sit ea ea do reprehenderit culpa duis. Nostrud

```

<div id="list-example" class="list-group">
  <a class="list-group-item list-group-item-action" href="#list-item-1">Item 1</a>
  <a class="list-group-item list-group-item-action" href="#list-item-2">Item2</a>
  <a class="list-group-item list-group-item-action" href="#list-item-3">Item 3</a>
  <a class="list-group-item list-group-item-action" href="#list-item-4">Item 4</a>
</div>

```

```
<div data-spy="scroll" data-target="#list-example" data-offset="0"
class="scrollspy-example">
  <h4 id="list-item-1">Item 1</h4>
  <p>...</p>
  <h4 id="list-item-2">Item 2</h4>
  <p>...</p>
  <h4 id="list-item-3">Item 3</h4>
  <p>...</p>
  <h4 id="list-item-4">Item 4</h4>
  <p>...</p>
</div>
```

用法

通过数据属性

要轻松添加滚动行为到您的顶栏导航, 添加 `data-spy="scroll"` 到您要窥探的元素(通常是`<body>`)。然后添加 `data-target` 属性到任何 Bootstrap `.nav` 组件的父元素 ID 或类的 Class 属性。

Copy

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

Copy

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

通过 JavaScript

在你的 CSS 加上 `position: relative;` , 通过 JavaScript 调用滚动监控:

Copy

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

需要可分辨（明确）的 ID 目标

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

忽略非:visible 目标元素

目标元素如果是非 :visible ，在 [jQuery 中将被忽略](#)，其它相对应的导航组永远不会被突出显示。

方法

.scrollspy('refresh')

当从 DOM 加入或删除元素使用滚动监控时，你需要调用刷新方式，如下所示：

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

.scrollspy('dispose')

销毁一个滚动元素。

选项

选项可以通过数据属性或 JavaScript 传递。对于数据属性，请将选项名称附加到 data- ,如 data-offset=""。

名称	Type 类型	默认值	描述
offset	number	10	计算滚动位置时，从顶部开始的计算的像偏移距离。

事件

Event 事件类型	描述
activate.bs.scrollspy	每当新项目被滚动激活时，该事件就会在滚动元素上触发。

```
Copy
$('[data-spy="scroll"]').on('activate.bs.scrollspy', function () {
    // do something...
})
```

提示冒泡 (Tooltip)

使用 CSS3 为 CSS 和 JavaScript 添加自定义 Bootstrap 工具提示的文档和示例，用于本地标题存储的动画和数据属性。

概览

使用提示冒泡持件时应了解以下：

- Tooltip 提示冒泡组件依赖于 [Popper.js](#)，你必须将 [popper.min.js](#) 文件放在 `bootstrap.js` 之前调用，或者使用 `bootstrap.bundle.min.js` / `bootstrap.bundle.js` 这两个已经包含了 `Popper.js` 的脚本。
- 如果自行编译 JS，请 [包含 util.js](#) 文件。
- 出于性能安排，Tooltip 提示冒泡是一个可选插件，所以 **你必须自己初始化它们**。
- 标题（或内容）零长度情况下，Tooltip 提示冒泡插件不会显示（生效）。
- 指定 `container: 'body'` 以避免复杂组年（如输入框组、按钮组等）中渲染呈出混乱（出问题）。
- 在隐藏元素上触发 Tooltip 提示冒泡插件不会起作用（无效行为）。
- Tooltip 提示冒泡插件的 `.disabled` 或 `disabled` 元素必须放在外层（父层）元素上。
- 如果从一个跨多行的链接上触发 Tooltip 提示冒泡插件，提示冒泡将居中，在你的 `<a>` 上使用 `white-space: nowrap`；可避免这种行为（即回归左右对齐）。
- 必须先隐藏 Tooltip 提示冒泡插件，然后才能从 DOM 中删除相应的元素。

都明白了？很好。让我们来通过一些例子来看它们具体是如何运作的。

示例：在任何地方启用 tooltip 提示冒泡插件

在网页上初始化所有的 tooltip 提示冒泡插件一个途径就是用 `data-toggle` 来选择它们：

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

实际范例

请用鼠标点击下面一段文字上的链接，查看 tooltip 提示冒泡效果：

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

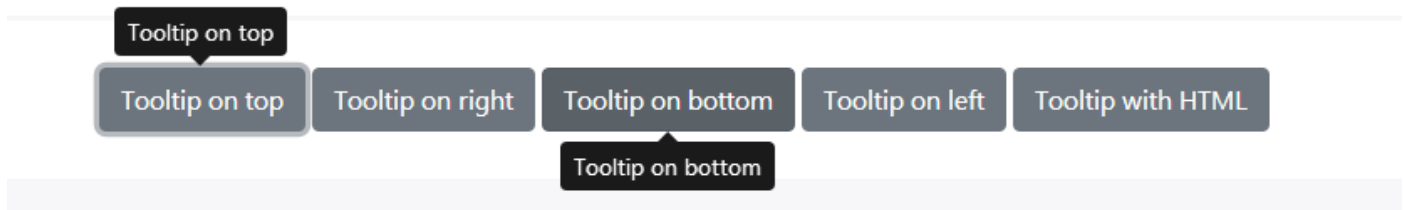
静态演示

可以使用四个方向：顶部对齐、右对齐、底部对齐、左对齐。



互动演示

用鼠标点击下列按钮可查看实际提示冒泡效果：



```
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="top" title="Tooltip on top">
  Tooltip on top
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="right" title="Tooltip on right">
  Tooltip on right
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="bottom" title="Tooltip on bottom">
  Tooltip on bottom
</button>
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-placement="left" title="Tooltip on left">
  Tooltip on left
</button>
```

而且支持自定义 HTML：

Copy

```
<button type="button" class="btn btn-secondary" data-toggle="tooltip" data-html="true" title="<em>Tooltip</em> <u>with</u> <b>HTML</b>">
  Tooltip with HTML
</button>
```

用法

tooltip 提示冒泡插件根据需要生成内容和标记，默认情况下我们将它放在触发元素之后。

使用 JavaScript 触发提示冒泡：

Copy

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


标记

工具提示框所需要的标记只是一个 `data` 元素和你希望拥有一个 tooltip 提示冒泡 HTML 元素上的 `title`，tooltip 提示冒泡插件的标记很简单，尽管它需要一个位置（默认是 `top` 顶部指示）。

使 tooltip 提示冒泡使用于键盘和辅助技术使用者

推荐你只为传统的键盘和互动式（如链接或表单控制元件）的 HTML 元素添加 tooltip 提示冒泡框，虽然任意的 HTML 元素 (如 ``) 可以通过添加 `tabindex="0"` 属性来调整 focus 行为，但这会为键盘使用者的非互动式元素增加可能困惑（混乱/错误)的定位点。此外大多数辅助技术目前还没有加入这种情况下的提示冒泡效果支持。

Copy

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

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

选项

你可以通过数据属性或 JavaScript 传递选项，如果使用数据属性，请将选项名称附加到 `data-`，如 `data-animation=""`。

名称	Type 类型	默认值	描述
animation	boolean	true	将 CSS 淡入淡出应用于 tooltip 提示冒泡。
container	string element false	false	将 tooltip 提示冒泡框附加到特定的元素，如 <code>container: 'body'</code> ，该选项特别有用，因为它允许您将触摸屏定位在触发元素附近的文字内容-这将防止在画面调整大小时，弹出的进示框远离触发元素。
delay	number object	0	显示和隐藏弹出提示框的延迟(ms)-不适用于手动触发类型。 如果向这个选项提供一个数字，隐藏/显示都会应用这个延迟。 对象结构是: <code>delay: { "show": 500, "hide": 100 }</code>
html	boolean	false	在 tooltip 提示冒泡中插入 HTML。 如果值为 true,tooltip 提示冒泡的 <code>title</code> 中 HTML 标签将在工具提示框中呈现。如果是 <code>false</code> ，则将使用 jQuery 的 <code>text</code> 方法将内容插入到 DOM 中。 如果你担心 XSS 攻击，请使用 <code>text</code> 文字。

名称	Type 类型	默认值	描述
placement	string function	'top'	<p>如何提示冒泡，包括： <code>auto top bottom left right</code>。如何指定为 <code>auto</code>，它会动态地调整提示冒泡的位置。举个例子，如果 <code>placement</code> 是 <code>auto left</code>，提示冒泡将尽可能地显示在左侧，否则显示在右侧。</p> <p>如果使用一个函数来定位时，将使用提示冒泡的 DMO 节点作为其第一个参数并将触发元素的 DOM 节点作为其第二个参数来调用，<code>this</code> 将被设为弹出提示框实例。</p>
selector	string false	false	<p>如果提供了选择器，提示冒泡将被委派给指定的目标。在实践中，它让动态添加的 HTML 内容也能附加提示框，看 这里 和 一个翔实的例子。</p>
template	string	'<div class="tooltip" role="tooltip"><div class="arrow"></div><div class="tooltip-inner"></div></div>'	<p>在创建提示冒泡时使用 HTML。</p> <p>tooltip 提示冒泡的 <code>title</code> 值将被注入到 <code>.tooltip-inner</code> 中。</p> <p><code>.arrow</code> 将变成 tooltip 提示冒泡的指示箭头。</p> <p>最外层的包裹元素应该有 <code>.tooltip</code> class 样式选择器。</p>
title	string element function	''	<p>如果 <code>title</code> 属性不存在，则提供默认 <code>title</code> 值。</p> <p>如果提供了一个函数，调用此函数时，此函数的 <code>this</code> 引用被设置为 tooltip 提示冒泡的元素。</p>
trigger	string	'hover focus'	<p>如何触发提示冒泡 - <code>click hover focus manual</code>，你可以传递多个触发器，用空格隔开它们，但是 <code>manual</code> 不能与别的触发器结合使用。</p>
offset	number string	0	<p>提示冒泡框对于其目标的偏移，更多信息请参阅 Popper.js 的 偏移（约束）文档。</p>
fallbackPlacement	string array	'flip'	<p>指定提示冒泡框在回调时使用哪个位置，有关更多信息请参阅 Popper.js 的 偏移（约束）文档。</p>

单个提示冒泡的数据属性

单个提示冒泡的选项可以通过使用 `data` 数据属性来替补指定，如前文所述。

方法

异步传输和转换

所有的 API 方法都是**异步**的，并开始一个转换。一旦转换开始但在结束之前，它们就返回给调用者。另外，一个转换组件的方法调用将被忽略。

[请参阅我们的 JavaScript 文档以获取更多信息。](#)

`$.tooltip(options)`

将一个元素附加一个提示冒泡处理程序。

`.tooltip('show')`

显示一个元素的提示冒泡，在提示冒泡真正显示之前返回给调用者 (即 `shown.bs.tooltip` 事件发生前)。这将被识别为一个“人为”的手动触发提示冒泡，零长度的提示框不会显示。

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

隐藏元素的冒泡提示，在提示冒泡框实际被隐藏之前返回给调用者 (即 `hidden.bs.tooltip` 事件发生前)。这将被识别为一个“人为”的手动触发提示冒泡。

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

切换元素的工具提示冒泡，在提示冒泡真正显示或隐藏之前返回给调用者 (即 `shown.bs.tooltip` 或 `hidden.bs.tooltip` 事件发生前)。这将被识别为一个“人为”的手动触发提示冒泡。

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

隐藏或破坏元素的提示冒泡，使用委派(创建时 [使用了 selector 选项](#))的提示冒泡不能在后代触发元素上单独销毁。

`$('#element').tooltip('dispose')` `.tooltip('enable')`

给一个元素的提示冒泡显示的功能，默认情况下启用 **Tooltip** 提示冒泡。

`$('#element').tooltip('enable')` `.tooltip('disable')`

删除元素的提示冒泡的显示功能，只有在重新启用后，才能显示提示冒泡。

`$('#element').tooltip('disable')` `.tooltip('toggleEnabled')`

切换元素的提示冒泡显示或隐藏的能力。

`$('#element').tooltip('toggleEnabled')` `.tooltip('update')`

更新 tooltip 提示冒泡的位置。

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

活动

Event 事件类型	描述
show.bs.tooltip	当调用 <code>show</code> 实例方法时，会立即触发该事件。
shown.bs.tooltip	当提示冒泡对用户来说可见时（需要等待 CSS 过渡完成），会触发该事件。
hide.bs.tooltip	当调用 <code>hide</code> 实例方法时，会立即触发该事件。
hidden.bs.tooltip	当提示冒泡对用户来说终于完成隐藏时（需要等待 CSS 过渡完成），会触发该事件。
inserted.bs.tooltip	将提示冒泡框加到 DOM 后，会在 <code>show.bs.tooltip</code> 事件后触发此事件。

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

公共样式

边框(border)

使用边框通用定义类来快速设置元素的边框和边框半径，适用于图像、按钮或任何其他元素。

基本边框

添加边框属性，显示指定边框。



```
<span class="border"></span>
<span class="border-top"></span>
<span class="border-right"></span>
<span class="border-bottom"></span>
<span class="border-left"></span>
```

在一个空间上定义边框-删除或显示特定边框定义方法。



```
<span class="border"></span>
<span class="border-0"></span>
<span class="border-top-0"></span>
<span class="border-right-0"></span>
<span class="border-bottom-0"></span>
<span class="border-left-0"></span>
```

边框颜色

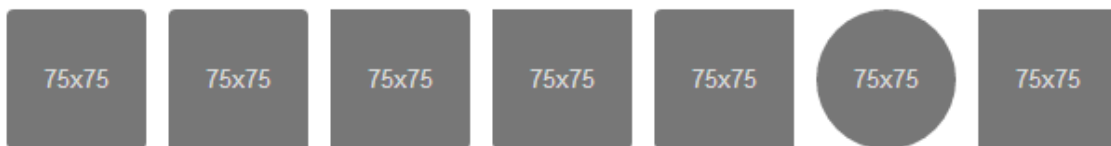
使用我们的主题颜色类方法，定义边框颜色。



```
<span class="border border-primary"></span>
<span class="border border-secondary"></span>
<span class="border border-success"></span>
<span class="border border-danger"></span>
<span class="border border-warning"></span>
<span class="border border-info"></span>
<span class="border border-light"></span>
<span class="border border-dark"></span>
<span class="border border-white"></span>
```

圆角边框

使用 `.rounded` 元素可以轻松的定义四个圆角的弧度及显示效果。



Copy

```







```

Clearfix 清动浮动

通过添加 `.clearfix` 实用程序，快速轻松地清除容器内浮动的内容（使元素换行呈现）。

`float` 类样式是通过添加 `.clearfix` 到父元素上来达到清除目标，也可以作为 Sass mixin 使用。

```
<div class="clearfix">...</div>
```

```
// Mixin itself @mixin clearfix() {  
  &::after {  
    display: block;  
    content: "";  
    clear: both;  
  }  
}
```

```
// Usage as a mixin  
.element {  
  @include clearfix;  
}
```

下面的实例展示了如何使用 `.clearfix`，没有 `.clearfix` 清除浮动，外层包裹的 DIV 不会被覆盖从而导致版位错乱。

Example Button floated left

Example Button floated right

```
<div class="bg-info clearfix">  
  <button type="button" class="btn btn-secondary float-left">Example Button  
floated left</button>  
  <button type="button" class="btn btn-secondary float-right">Example Button  
floated right</button>  
</div>
```

关闭图标

使用通用的 close 关闭图标来关闭 modals 模态框提示或 alert 提示组件的内容。

可使用 `aria-label` 标签，为屏幕阅读器用户添加文字定义（为阅读障碍者提供访问支持）。

```
×  
<button type="button" class="close" aria-label="Close">  
  <span aria-hidden="true">&times;</span>  
</button>
```

颜色

通过颜色传达意义、表达不同的模块，这有一系列的定义方法，包括支持链接、悬停、选中等状态相关的的样式集。

`.text-primary`

`.text-secondary`

`.text-success`

`.text-danger`

`.text-warning`

`.text-info`

`.text-light`

`.text-dark`

`.text-muted`

`.text-white`

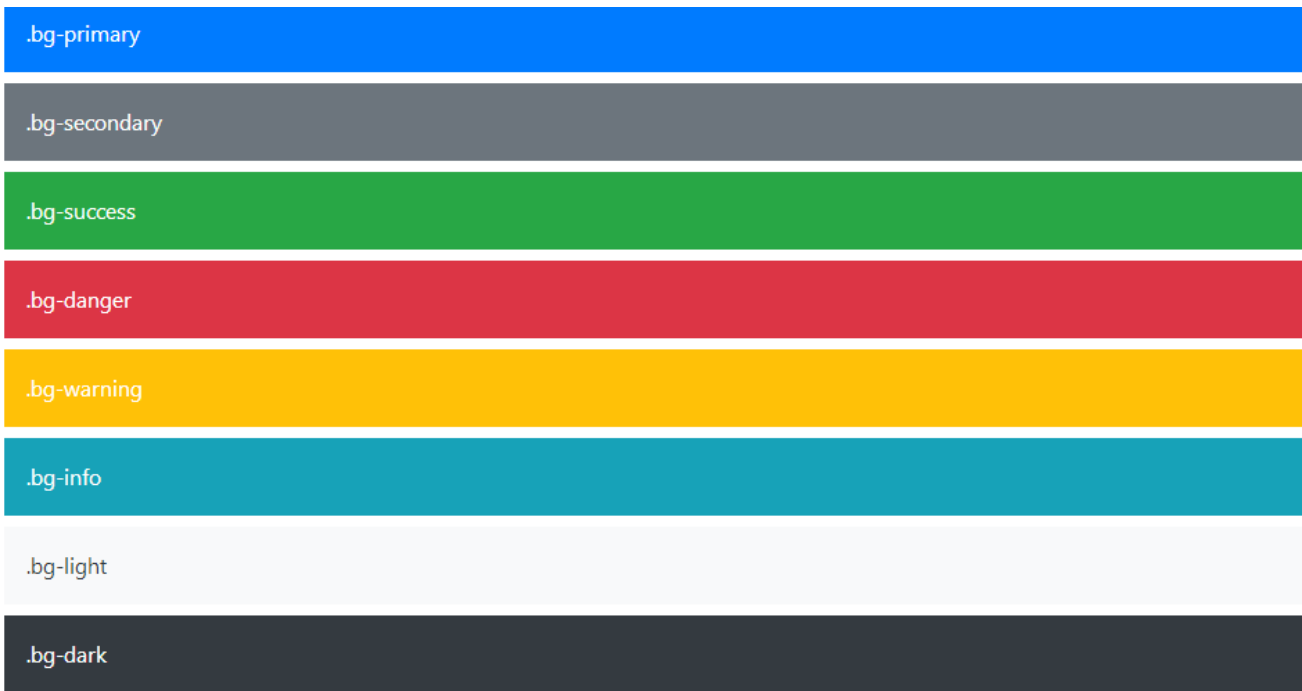
```
<p class="text-primary">.text-primary</p>
<p class="text-secondary">.text-secondary</p>
<p class="text-success">.text-success</p>
<p class="text-danger">.text-danger</p>
<p class="text-warning">.text-warning</p>
<p class="text-info">.text-info</p>
<p class="text-light bg-dark">.text-light</p>
<p class="text-dark">.text-dark</p>
<p class="text-muted">.text-muted</p>
<p class="text-white bg-dark">.text-white</p>
```

使用我们提供的悬停和焦点状态（情景）样式，在链接上也能正常使用（呈现），**注意：**`.text-white`、`.text-muted` 这两个 `class` 样式不支持链接上使用（没有链接样式）。

```
<p><a href="#" class="text-primary">Primary link</a></p>
<p><a href="#" class="text-secondary">Secondary link</a></p>
<p><a href="#" class="text-success">Success link</a></p>
<p><a href="#" class="text-danger">Danger link</a></p>
<p><a href="#" class="text-warning">Warning link</a></p>
<p><a href="#" class="text-info">Info link</a></p>
<p><a href="#" class="text-light bg-dark">Light link</a></p>
<p><a href="#" class="text-dark">Dark link</a></p>
<p><a href="#" class="text-muted">Muted link</a></p>
<p><a href="#" class="text-white bg-dark">White link</a></p>
```

背景颜色

与 text 文字类颜色 class 定义相同，链接元互会在 hover 状态时变暗。背景色 **不要设置** `color` 样式，尽可能使用 `.text-*` 通用 CSS 类。



.bg-white

```
<div class="p-3 mb-2 bg-primary text-white">.bg-primary</div>
<div class="p-3 mb-2 bg-secondary text-white">.bg-secondary</div>
<div class="p-3 mb-2 bg-success text-white">.bg-success</div>
<div class="p-3 mb-2 bg-danger text-white">.bg-danger</div>
<div class="p-3 mb-2 bg-warning text-white">.bg-warning</div>
<div class="p-3 mb-2 bg-info text-white">.bg-info</div>
<div class="p-3 mb-2 bg-light text-dark">.bg-light</div>
<div class="p-3 mb-2 bg-dark text-white">.bg-dark</div>
<div class="p-3 mb-2 bg-white text-dark">.bg-white</div>
```

背景渐变

当 `$enable-gradients` 设置为 `true`，则可以使用 `.bg-gradient-` 通用样式。默认情况下，`$enable-gradients` 是被禁用的，如同下面的示例也是故意被破坏显示错误的。这是为了在您使用 Bootstrap 时更加方便的进行自定义，[了解我们的 Sass 选项](#) 以启用这些 Class 及更多。

```
<div class="p-3 mb-2 bg-gradient-primary text-white">.bg-gradient-primary</div>
<div class="p-3 mb-2 bg-gradient-secondary text-white">.bg-gradient-secondary</div>
<div class="p-3 mb-2 bg-gradient-success text-white">.bg-gradient-success</div>
<div class="p-3 mb-2 bg-gradient-danger text-white">.bg-gradient-danger</div>
<div class="p-3 mb-2 bg-gradient-warning text-white">.bg-gradient-warning</div>
<div class="p-3 mb-2 bg-gradient-info text-white">.bg-gradient-info</div>
<div class="p-3 mb-2 bg-gradient-light text-dark">.bg-gradient-light</div>
<div class="p-3 mb-2 bg-gradient-dark text-white">.bg-gradient-dark</div>
```

Display 显示属性

通过我们的显示实用程序，可以快速、有效地切换组件的显示值和更多，包括对一些更常见的值的支持，以及一些用于在打印时控制显示的附加功能。

Display 运行原理

使用我们的响应式 `display 实用` 程序类更改显示属性的值。系统默认特意支持 `display` 所有可能值的一个子集。

Dislay 属性

显示适用于从 `xs` 到 `xl` 的所有 `breakpoints` 的实用程序类, 其中没有 `breakpoints` 缩写。这是因为这些类是从 `min-width: 0;` 并且因此不受媒体查询的限制。 然而，其余的断点包含断点缩写。

因此，这些类使用以下格式来命名：

- `.d-{value}` 用于小屏幕适配（即手机适配）
- `.d-{breakpoint}-{value}` 用于 `sm`、`md`、`lg`、`xl` 等多设备适配。

`display` 常用属性：

- `none`
- `inline`
- `inline-block`
- `block`
- `table`
- `table-cell`
- `table-row`
- `flex`
- `inline-flex`

媒体查询效果屏幕宽度与给定的断点或更大。 例如，`.d-lg-none sets display: none;` 在 `lg` 和 `xl` 屏幕上。

实例



```
<div class="d-inline p-2 bg-primary text-white">d-inline</div>
<div class="d-inline p-2 bg-dark text-white">d-inline</div>
```

d-block

d-block

```
<span class="d-block p-2 bg-primary text-white">d-block</span>
<span class="d-block p-2 bg-dark text-white">d-block</span>
```

隐藏的元素

为了更快速且友好 的支持移动设备开发，请使用 `display classes` 来显示和隐藏组件，避免创建完全不同版本的一个网站（为移动网站建立一个独立的站点），而不是按照每种屏幕尺寸来隐藏元素。

隐藏元素只要使用 `.d-none` class 或 `.d-{sm,md,lg,xl}-none` 的任何变量来支持响应式。

如要在指定的屏幕上显示一个元素，则可以将一个 `.d-*-none` class 样式与 `.d-*-*` class 样式结合起来，如 `.d-none.d-md-block.d-xl-none` 将隐藏除了中型、大型设备以外的所有屏幕中的元素，。

屏幕规格	引用样式
所有屏幕下隐藏	<code>.d-none</code>
只在 xs 屏幕上隐藏（即仅在手机屏幕上隐藏-其它规格屏幕正常显示-译者注）	<code>.d-none .d-sm-block</code>
只在 sm 屏幕上隐藏（其它屏幕规格均显示）	<code>.d-sm-none .d-md-block</code>
只在 md 屏幕时隐藏（其它屏幕规格均显示）	<code>.d-md-none .d-lg-block</code>
只在 lg 屏幕时隐藏（其它屏幕规格均显示）	<code>.d-lg-none .d-xl-block</code>
只在 xl 屏幕时隐藏（其它屏幕规格均显示）	<code>.d-xl-none</code>
全部可见	<code>.d-block</code>
仅在 xs 屏幕时可见	<code>.d-block .d-sm-none</code>
仅在 sm 屏幕时可见	<code>.d-none .d-sm-block .d-md-none</code>
仅在 md 屏幕时可见	<code>.d-none .d-md-block .d-lg-none</code>
仅在 lg 屏幕时可见	<code>.d-none .d-lg-block .d-xl-none</code>
仅在 xl 屏幕时可见	<code>.d-none .d-xl-block</code>

hide on screens smaller than lg

```
<div class="d-lg-none">hide on screens wider than lg</div>
```

```
<div class="d-none d-lg-block">hide on screens smaller than lg</div>
```

面向打印的显示属性控制处理

在处理打印样式时，通过 `display` 通用样式来改变相应值处理呈现效果。

- `.d-print-none`
- `.d-print-inline`
- `.d-print-inline-block`
- `.d-print-block`
- `.d-print-table`
- `.d-print-table-row`
- `.d-print-table-cell`
- `.d-print-flex`
- `.d-print-inline-flex`

印刷和显示课程可以合并。

Screen Only (Hide on print only)

Hide up to large on screen, but always show on print

```
<div class="d-print-none">Screen Only (Hide on print only)</div>
```

```
<div class="d-none d-print-block">Print Only (Hide on screen only)</div>
<div class="d-none d-lg-block d-print-block">Hide up to large on screen, but always
show on print</div>
```

嵌入

创建响应式的视频、图像、幻灯片，并能在任何设备上友好的扩展显示。

关于

将这些规则应用到 `<iframe>`、`<embed>`、`<video>`、`<object>` 上，当需要配合其它属性（如响应式）时，也可以加入 `.embed-responsive-item` 定义。

小技七! 你不需要将 `frameborder="0"` 加入到你的 `<iframe>` 中，因为我们已经为您覆盖处理了这个属性。

示例

你可以将任何代码嵌入到 `<iframe>` 中，并使用 `.embed-responsive` 实现同比例收缩，至于 `.embed-responsive-item` 不是严格要求的-虽然我们鼓励使用它。

```
<div class="embed-responsive embed-responsive-16by9">
  <iframe class="embed-responsive-item"
src="https://www.youtube.com/embed/zp0ULjyy-n8?rel=0" allowfullscreen></iframe>
</div>
```

长宽比例处理

可以通过修改 class 样式来实现纵横比定义，如下：

Copy

```
<!-- 21:9 aspect ratio -->
<div class="embed-responsive embed-responsive-21by9">
  <iframe class="embed-responsive-item" src="..."></iframe>
</div>

<!-- 16:9 aspect ratio -->
<div class="embed-responsive embed-responsive-16by9">
  <iframe class="embed-responsive-item" src="..."></iframe>
</div>

<!-- 4:3 aspect ratio -->
<div class="embed-responsive embed-responsive-4by3">
  <iframe class="embed-responsive-item" src="..."></iframe>
</div>

<!-- 1:1 aspect ratio -->
```

```
<div class="embed-responsive embed-responsive-1by1">
  <iframe class="embed-responsive-item" src="..."></iframe>
</div>
```

Flex 弹性布局

引入新的 Flex 弹性布局，可以实现通过一整套响应灵活的实用程序，快速管理栅格的列、导航、组件等的布局、对齐和大小。通过进一步的定义 CSS，还可以实现更复杂的展示样式。

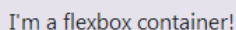
启用弹性行为

使用 `display` 通用属性来创建一个 flexbox 容器，并将 **直属内部子元素** 转换为 flex 属性。flex 元素的容器和子项目可以通过额外的 flex 属性定义来实现进一步修改。

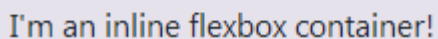
FlexBox 布局提供了包括诸多优秀特性，其中包括：

1. 在父元素里面内容的简单的垂直对齐
2. 通过使用媒体查询，可以简单的根据设备和屏幕分辨率来对内容重新排序
3. CSS 就能实现等高列布局以及垂直处理

中文版的 FlexBox 布局教程可见：<https://www.z01.com/help/web/3234.shtml>（译者注）



```
<div class="d-flex p-2">I'm a flexbox container!</div>
```



```
<div class="d-inline-flex p-2">I'm an inline flexbox container!</div>
```

响应式变化也存在于 `.d-flex` and `.d-inline-flex` 这两个通用 Class 上。

- `.d-flex`
- `.d-inline-flex`
- `.d-sm-flex`
- `.d-sm-inline-flex`
- `.d-md-flex`
- `.d-md-inline-flex`
- `.d-lg-flex`
- `.d-lg-inline-flex`
- `.d-xl-flex`
- `.d-xl-inline-flex`

方向

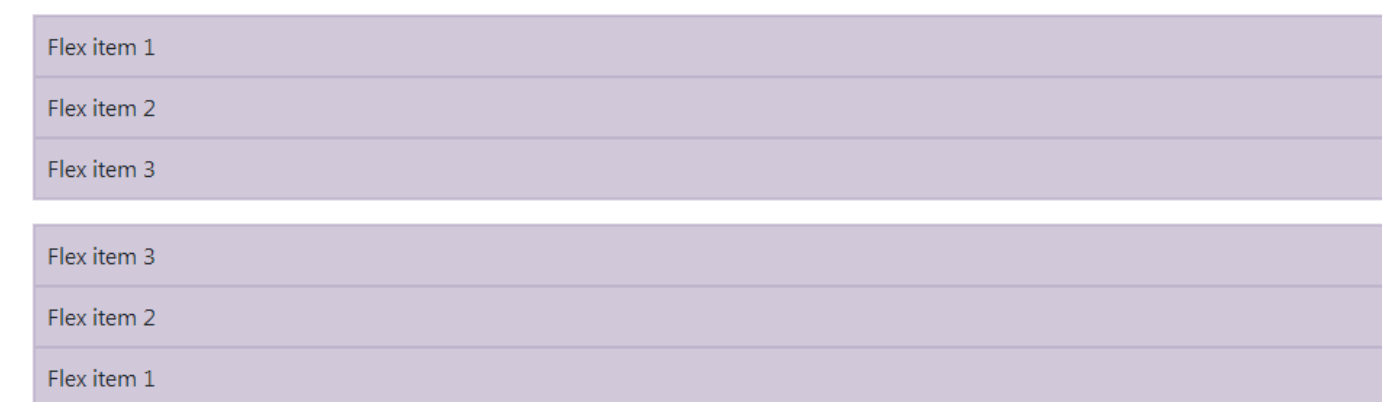
基于系统提供的通用样式定义，可以设定 flex 的容器与内部 flex 元素的方向，在大多数情况下你可以忽略水平的 class 样式定义，因为浏览器的默认值是 `row` 但在比如响应式布局时，可能需要进行显式设定此值。

浏览器预默认值下，使用 `.flex-row` 可设置子元素水平方向排版呈现，或者使用 `.flex-row-reverse` 可实现元素在水平上作反方向排列（右对齐、从右到左布局）。



```
<div class="d-flex flex-row">
  <div class="p-2">Flex item 1</div>
  <div class="p-2">Flex item 2</div>
  <div class="p-2">Flex item 3</div>
</div>
<div class="d-flex flex-row-reverse">
  <div class="p-2">Flex item 1</div>
  <div class="p-2">Flex item 2</div>
  <div class="p-2">Flex item 3</div>
</div>
```

使用 `.flex-column` 设置垂直方向布局，或使用 `.flex-column-reverse` 实现垂直方向的反转布局（从底向上铺开）。



```
<div class="d-flex flex-column">
  <div class="p-2">Flex item 1</div>
  <div class="p-2">Flex item 2</div>
  <div class="p-2">Flex item 3</div>
</div>
<div class="d-flex flex-column-reverse">
  <div class="p-2">Flex item 1</div>
  <div class="p-2">Flex item 2</div>
  <div class="p-2">Flex item 3</div>
</div>
```

`flex-direction` 的响应式属性规范：

- `.flex-row`
- `.flex-row-reverse`

- `.flex-column`
- `.flex-column-reverse`
- `.flex-sm-row`
- `.flex-sm-row-reverse`
- `.flex-sm-column`
- `.flex-sm-column-reverse`
- `.flex-md-row`
- `.flex-md-row-reverse`
- `.flex-md-column`
- `.flex-md-column-reverse`
- `.flex-lg-row`
- `.flex-lg-row-reverse`
- `.flex-lg-column`
- `.flex-lg-column-reverse`
- `.flex-xl-row`
- `.flex-xl-row-reverse`
- `.flex-xl-column`
- `.flex-xl-column-reverse`

内容对齐与对准

使用 flexbox 弹性布局容器上的 `justify-content-*` 通用样式可以改变 flex 项目在主轴上的对齐（x 轴开始，如果 `flex-direction: column` 则为 y 轴），或选方向（值）包括：`start` (浏览器默认值)、`end`、`center`、`between`、`around`。



```
<div class="d-flex justify-content-start">...</div>
<div class="d-flex justify-content-end">...</div>
<div class="d-flex justify-content-center">...</div>
<div class="d-flex justify-content-between">...</div>
<div class="d-flex justify-content-around">...</div>
```

`justify-content-*` 样式也通用存在响应式属性规范：

- `.justify-content-start`
- `.justify-content-end`
- `.justify-content-center`
- `.justify-content-between`
- `.justify-content-around`

- `.justify-content-sm-start`
- `.justify-content-sm-end`
- `.justify-content-sm-center`
- `.justify-content-sm-between`
- `.justify-content-sm-around`
- `.justify-content-md-start`
- `.justify-content-md-end`
- `.justify-content-md-center`
- `.justify-content-md-between`
- `.justify-content-md-around`
- `.justify-content-lg-start`
- `.justify-content-lg-end`
- `.justify-content-lg-center`
- `.justify-content-lg-between`
- `.justify-content-lg-around`
- `.justify-content-xl-start`
- `.justify-content-xl-end`
- `.justify-content-xl-center`
- `.justify-content-xl-between`
- `.justify-content-xl-around`

对齐项目

使用 `align-items-*` 通用样式可以在 `flexbox` 容器上实现 `flex` 项目的对齐（y 轴开始，如果选择 `flex-direction: column` 则从 x 轴开始），可选参数有：`start`、`end`、`center`、`baseline`、`stretch`（浏览器默认值）。



```
<div class="d-flex align-items-start">...</div>
```



```
<div class="d-flex align-items-end">...</div>
<div class="d-flex align-items-center">...</div>
<div class="d-flex align-items-baseline">...</div>
<div class="d-flex align-items-stretch">...</div>
```

`align-items` 的响应式属性规范:

- `.align-items-start`
- `.align-items-end`
- `.align-items-center`
- `.align-items-baseline`
- `.align-items-stretch`
- `.align-items-sm-start`
- `.align-items-sm-end`
- `.align-items-sm-center`
- `.align-items-sm-baseline`
- `.align-items-sm-stretch`
- `.align-items-md-start`
- `.align-items-md-end`
- `.align-items-md-center`
- `.align-items-md-baseline`
- `.align-items-md-stretch`
- `.align-items-lg-start`
- `.align-items-lg-end`
- `.align-items-lg-center`
- `.align-items-lg-baseline`
- `.align-items-lg-stretch`
- `.align-items-xl-start`
- `.align-items-xl-end`
- `.align-items-xl-center`
- `.align-items-xl-baseline`
- `.align-items-xl-stretch`

自对齐

使用 `align-self-*` 通用样式可以使用 flexbox 上的项目单独改变在横轴上的对齐方式 (y 值开始, 如果 `flex-direction: column` 则为 x 轴开始), 其拥有与 `align-items` 相同的可选子项: `start`、`end`、`center`、`baseline`、`orstretch` (浏览器默认值)。

Flex item	Aligned flex item	Flex item	
Flex item		Flex item	
	Aligned flex item		
Flex item		Flex item	
	Aligned flex item		
Flex item	Aligned flex item	Flex item	
Flex item	Aligned flex item	Flex item	

```

<div class="align-self-start">Aligned flex item</div>
<div class="align-self-end">Aligned flex item</div>
<div class="align-self-center">Aligned flex item</div>
<div class="align-self-baseline">Aligned flex item</div>
<div class="align-self-stretch">Aligned flex item</div>

```

`align-self` 的响应式属性规范:

- `.align-self-start`
- `.align-self-end`
- `.align-self-center`
- `.align-self-baseline`
- `.align-self-stretch`
- `.align-self-sm-start`
- `.align-self-sm-end`
- `.align-self-sm-center`
- `.align-self-sm-baseline`
- `.align-self-sm-stretch`
- `.align-self-md-start`
- `.align-self-md-end`
- `.align-self-md-center`
- `.align-self-md-baseline`
- `.align-self-md-stretch`

- `.align-self-lg-start`
- `.align-self-lg-end`
- `.align-self-lg-center`
- `.align-self-lg-baseline`
- `.align-self-lg-stretch`
- `.align-self-xl-start`
- `.align-self-xl-end`
- `.align-self-xl-center`
- `.align-self-xl-baseline`
- `.align-self-xl-stretch`

自浮动 Auto margins

当你将 flex 对齐与 auto margin 混在一起的时候，flexbox 也能正常运行。以下是通过自动 margin 来控制 flex 项目的三种示例，分别是预设（无 margin）、向右推两个项目（`.mr-auto`）、向左推两个项目（`.ml-auto`）：

不幸的是，**IE10** 和 **IE11** 不能正确支持在父层具有非默认的 `justify-content` 值自边距浮动 **auto margin**，[可查阅 StackOverflow 对此现象的解答](#) 了解更多细节。

Flex item	Flex item	Flex item	
Flex item			Flex item Flex item
Flex item	Flex item		Flex item

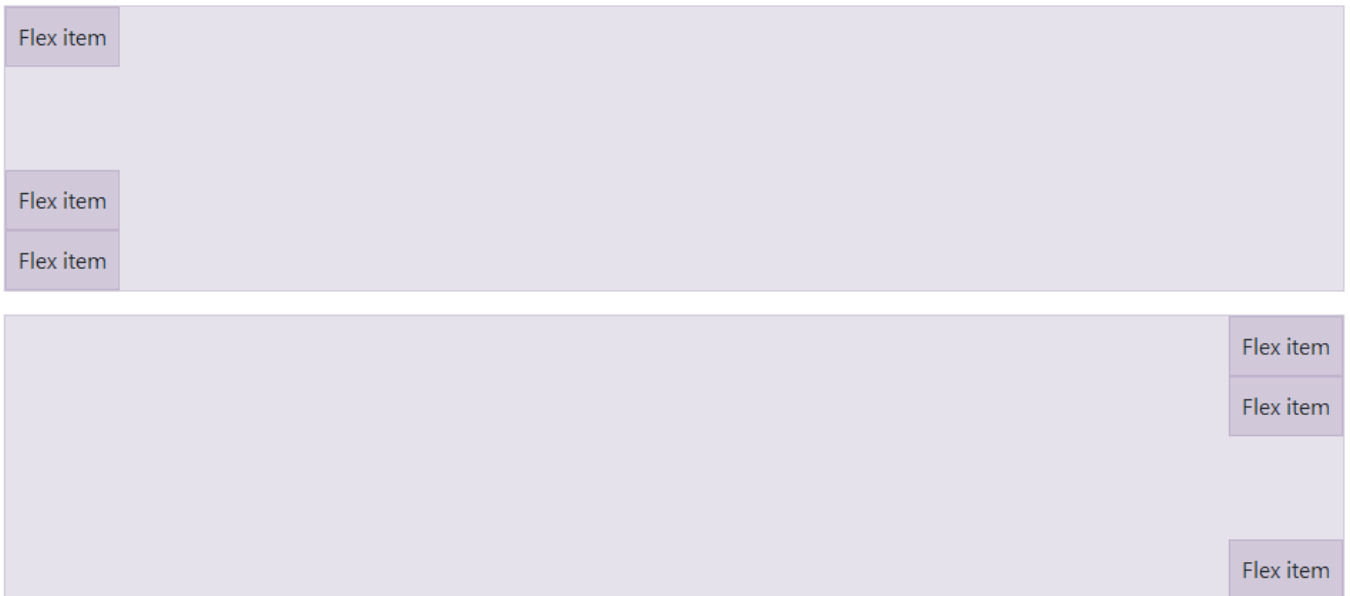
```
<div class="d-flex">
  <div class="p-2">Flex item</div>
  <div class="p-2">Flex item</div>
  <div class="p-2">Flex item</div>
</div>

<div class="d-flex">
  <div class="mr-auto p-2">Flex item</div>
  <div class="p-2">Flex item</div>
  <div class="p-2">Flex item</div>
</div>

<div class="d-flex">
  <div class="p-2">Flex item</div>
  <div class="p-2">Flex item</div>
  <div class="ml-auto p-2">Flex item</div>
</div>
```

与 align-items 结合实现垂直布局

结合 `align-items`、`flex-direction: column`、`margin-top: auto` 或 `margin-bottom: auto`，可以垂直移动一个 flex 子容器到顶部或底部。

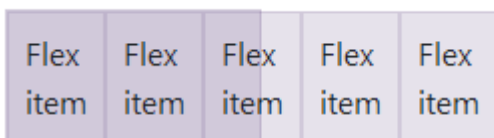


```
<div class="d-flex align-items-start flex-column" style="height: 200px;">
  <div class="mb-auto p-2">Flex item</div>
  <div class="p-2">Flex item</div>
  <div class="p-2">Flex item</div>
</div>
```

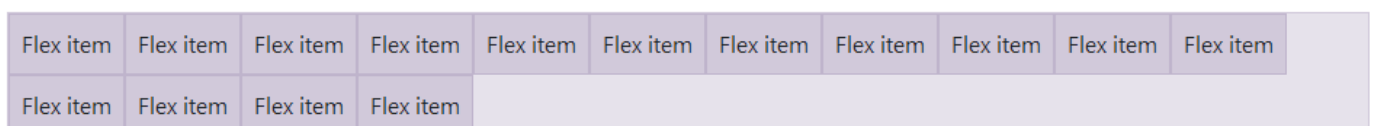
```
<div class="d-flex align-items-end flex-column" style="height: 200px;">
  <div class="p-2">Flex item</div>
  <div class="p-2">Flex item</div>
  <div class="mt-auto p-2">Flex item</div>
</div>
```

Wrap 包裹

改变 flex 项目在 flex 容器中的包裹方式（可以实现弹性布局），其中包括无包裹 `.flex-nowrap`（浏览器默认）、包裹 `.flex-wrap`，或者反向包裹 `.flex-wrap-reverse`。



```
<div class="d-flex flex-nowrap">
  ...
</div>
```



```
<div class="d-flex flex-wrap">
```

```
...
</div>
```

Flex item	Flex item	Flex item	Flex item							
Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item

```
<div class="d-flex flex-wrap-reverse">
```

```
...
</div>
```

`flex-*-wrap-*`的响应式规范:

- `.flex-nowrap`
- `.flex-wrap`
- `.flex-wrap-reverse`
- `.flex-sm-nowrap`
- `.flex-sm-wrap`
- `.flex-sm-wrap-reverse`
- `.flex-md-nowrap`
- `.flex-md-wrap`
- `.flex-md-wrap-reverse`
- `.flex-lg-nowrap`
- `.flex-lg-wrap`
- `.flex-lg-wrap-reverse`
- `.flex-xl-nowrap`
- `.flex-xl-wrap`
- `.flex-xl-wrap-reverse`

Order 排序

使用系统提供的通用样式定义可以实现弹性项目的 *可视化* 排序。我们仅提供将一个特件排在第一或最后, 以及重置 DOM 顺序, 由于 `order` 只能使用整数值 (如: `5`), 因此对于需要的任何额外值需要自定义 CSS。

Third flex item	Second flex item	First flex item	
-----------------	------------------	-----------------	--

```
<div class="d-flex flex-nowrap">
  <div class="order-3 p-2">First flex item</div>
  <div class="order-2 p-2">Second flex item</div>
  <div class="order-1 p-2">Third flex item</div>
</div>
```

`order-*`排序方法的响应式属性:

- `.order-1`
- `.order-2`
- `.order-3`
- `.order-4`
- `.order-5`

- .order-6
- .order-7
- .order-8
- .order-9
- .order-10
- .order-11
- .order-12
- .order-sm-1
- .order-sm-2
- .order-sm-3
- .order-sm-4
- .order-sm-5
- .order-sm-6
- .order-sm-7
- .order-sm-8
- .order-sm-9
- .order-sm-10
- .order-sm-11
- .order-sm-12
- .order-md-1
- .order-md-2
- .order-md-3
- .order-md-4
- .order-md-5
- .order-md-6
- .order-md-7
- .order-md-8
- .order-md-9
- .order-md-10
- .order-md-11
- .order-md-12
- .order-lg-1
- .order-lg-2
- .order-lg-3
- .order-lg-4
- .order-lg-5
- .order-lg-6
- .order-lg-7
- .order-lg-8
- .order-lg-9
- .order-lg-10
- .order-lg-11
- .order-lg-12
- .order-xl-1
- .order-xl-2
- .order-xl-3
- .order-xl-4
- .order-xl-5
- .order-xl-6

- `.order-xl-7`
- `.order-xl-8`
- `.order-xl-9`
- `.order-xl-10`
- `.order-xl-11`
- `.order-xl-12`

对齐内容

使用 flexbox 容器上的 `align-content` 通用样式定义，可以将 flex 物价于横轴上 *一起对齐*，可选方向有 `start` (浏览器默认值)、`end`、`center`、`between`、`around`、`stretch`。为了展现这些效果，下面实例我们已经实施了 `flex-wrap: wrap` 定义并增加了很多子项目的数量，如下所示：

注意! 此属性对于单行的 Flex 项目没有影响。

Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item
Flex item	Flex item	Flex item	Flex item							

```
<div class="d-flex align-content-start flex-wrap">
  ...
</div>
```

Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item
Flex item	Flex item	Flex item	Flex item							

```
<div class="d-flex align-content-end flex-wrap">...</div>
```

Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item
Flex item	Flex item	Flex item	Flex item							

```
<div class="d-flex align-content-center flex-wrap">...</div>
```

Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item
Flex item	Flex item	Flex item	Flex item							

```
<div class="d-flex align-content-between flex-wrap">...</div>
```

Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item
Flex item	Flex item	Flex item	Flex item							

```
<div class="d-flex align-content-around flex-wrap">...</div>
```

Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item	Flex item
Flex item	Flex item	Flex item	Flex item							

```
<div class="d-flex align-content-stretch flex-wrap">...</div>
```

`align-content-*`方法的响应式属性:

- `.align-content-start`
- `.align-content-end`
- `.align-content-center`
- `.align-content-around`
- `.align-content-stretch`
- `.align-content-sm-start`
- `.align-content-sm-end`
- `.align-content-sm-center`
- `.align-content-sm-around`
- `.align-content-sm-stretch`
- `.align-content-md-start`
- `.align-content-md-end`
- `.align-content-md-center`
- `.align-content-md-around`
- `.align-content-md-stretch`
- `.align-content-lg-start`
- `.align-content-lg-end`
- `.align-content-lg-center`
- `.align-content-lg-around`
- `.align-content-lg-stretch`

- `.align-content-xl-start`
- `.align-content-xl-end`
- `.align-content-xl-center`
- `.align-content-xl-around`
- `.align-content-xl-stretch`

Float 浮动属性

使用我们的响应式 `float` 浮动通用样式，能在任何设备断点（浏览器尺寸）上切换浮动。

概览

这些通用样式定义可定义元素浮动到左侧或右侧，或者 使用 [CSS float 属性](#)实现基于当前浏览器窗口禁用浮动。可引入 `!important` 来避免特殊 hack，这些属性都使用 Bootstrap 全局栅格相当的设备断点（屏幕规格定义）。

Class 样式

使用 class 样式来切换 float 效果:

在所有 viewport 窗口上浮动在左侧

在所有 viewport 窗口上浮动到右侧

所有 viewport 窗口都不浮动

```
<div class="float-left">在所有 viewport 窗口上浮动在左侧</div><br>
<div class="float-right">在所有 viewport 窗口上浮动到右侧</div><br>
<div class="float-none">所有 viewport 窗口都不浮动</div>
```

Mixin 引导

或者使用 Sass mixin:

```
.element {
  @include float-left;
}
.another-element {
  @include float-right;
}
.one-more {
  @include float-none;
}
```

响应式可选参数

`.float-*` 的响应式属性规范:

Float left on viewports sized SM (small) or wider

Float left on viewports sized MD (medium) or wider

Float left on viewports sized LG (large) or wider

Float left on viewports sized XL (extra-large) or wider

```
<div class="float-sm-left">Float left on viewports sized SM (small) or wider</div><br>
<div class="float-md-left">Float left on viewports sized MD (medium) or wider</div><br>
<div class="float-lg-left">Float left on viewports sized LG (large) or wider</div><br>
<div class="float-xl-left">Float left on viewports sized XL (extra-large) or wider</div><br>
```

这是所有子 class 类定义：

- `.float-left`
- `.float-right`
- `.float-none`
- `.float-sm-left`
- `.float-sm-right`
- `.float-sm-none`
- `.float-md-left`
- `.float-md-right`
- `.float-md-none`
- `.float-lg-left`
- `.float-lg-right`
- `.float-lg-none`
- `.float-xl-left`
- `.float-xl-right`
- `.float-xl-none`

图像替换

使用图像替换 class 样式类，实现文字与背景图的替换。

使用 `.text-hide` class 样或 sass mixin 来隐藏一个元素的文字内容并替换成背景图片。

```
<h1 class="text-hide">Custom heading</h1>
// Usage as a mixin .heading {
  @include text-hide;
}
```

使用 `.text-hide` class 样式可以保持标签的亲及性及 SEO 优化需求，引入后，需要使用 `background-image` 属性来提供视觉展示，而不是文字内容（文字内容随即隐藏）。

```
<h1 class="text-hide" style="background-image:
url('/assets/brand/bootstrap-solid.svg'); width: 50px; height:
50px;">Bootstrap</h1>
```

固顶(底)及定位

使用下面这些样式方法可以快速对元素的位置进行设定，包括固顶、固底以及定位。

固顶(底)及定位

使用下面这些样式方法可以快速对元素的位置进行设定，包括固顶、固底以及定位。

```
<div class="position-static">...</div>
<div class="position-relative">...</div>
<div class="position-absolute">...</div>
<div class="position-fixed">...</div>
<div class="position-sticky">...</div>
```

固定在 top 顶部

将一个元素固定在可见区域的顶部，从边到边的对齐，用户在使用固定在顶部时请确认效果带来的影响（如覆盖）-必要时增加额外的自定义 CSS。

```
<div class="fixed-top">...</div>
```

固定在 bottom 底部

将一个元素固定在可见区域的底部，从边到边的对齐，用户在使用固定在底部时请确认效果带来的影响（如覆盖）-必要时增加额外的自定义 CSS。

```
<div class="fixed-bottom">...</div>
```

贴齐于 top 顶部

将一个元素贴齐于可见区域的顶部，从边到边-但只在你的浏览器窗口滚动才能激活它，该 `.sticky-top` 样式使用 `position: sticky` 不能在所有浏览器中获得支持。

Microsoft Edge 和 IE11 呈现 `position: sticky` 使用的是 `position: relative` 属性，因此我们将这个样式增加了 `@supports` 动态变量，限制在可支持的浏览器上运行。

```
<div class="sticky-top">...</div>
```

规格与尺寸(Sizing)

使用系统宽度和高度样式，轻松地定义任何元素的宽或高（相对于其父级）。

宽度和高度通用属性从 `_variables.scss` 的 `$sizes` Sass map 中产生，默认值包括 `25%`、`50%`、`75%`、`100%`，你可根据整这些值，从而生产出不同的规格属性。

Width 25%

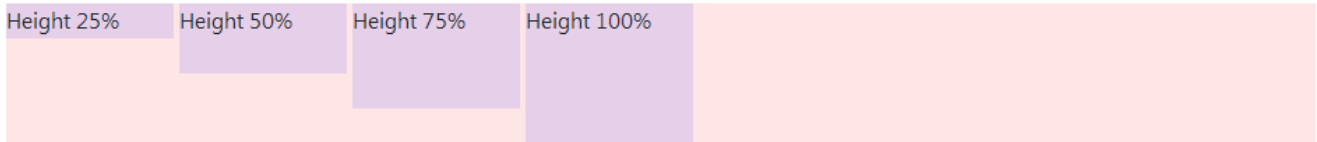
Width 50%

Width 75%

Width 100%

```
<div class="w-25 p-3" style="background-color: #eee;">Width 25%</div>
<div class="w-50 p-3" style="background-color: #eee;">Width 50%</div>
<div class="w-75 p-3" style="background-color: #eee;">Width 75%</div>
```

```
<div class="w-100 p-3" style="background-color: #eee;">Width 100%</div>
```



```
<div style="height: 100px; background-color: rgba(255,0,0,0.1);">
  <div class="h-25 d-inline-block" style="width: 120px; background-color:
  rgba(0,0,255,.1)">Height 25%</div>
  <div class="h-50 d-inline-block" style="width: 120px; background-color:
  rgba(0,0,255,.1)">Height 50%</div>
  <div class="h-75 d-inline-block" style="width: 120px; background-color:
  rgba(0,0,255,.1)">Height 75%</div>
  <div class="h-100 d-inline-block" style="width: 120px; background-color:
  rgba(0,0,255,.1)">Height 100%</div>
</div>
```

你也可使用 `max-width: 100%;` 和 `max-height: 100%;` 这些通用样式定义。

```

```

```
<div style="height: 100px; background-color: rgba(255,0,0,0.1);">
  <div class="mh-100" style="width: 100px; height: 200px; background-color:
  rgba(0,0,255,0.1);">Max-height 100%</div>
</div>
```

spacing 间隔规范（Margin 与 Padding 间距处理）

Bootstrap 4 内置了各种的快速缩进、隔离、填充等间距处理工具，响应余量和填充实用程序类来修改元素的外观。

工作原理

系统提供了一组缩写 CSS，并赋予 `margin` 或 `padding` 值，包括对单个属性、所有 属性以及垂直、水平等属性的支持，距离单位是在采用 `0.25rem` 到 `3rem`，Class 来源于 Sass map 定义。

缩写 CSS 符约定

Spacing 通用样式适用于所有屏幕尺寸，从 `xs` 到 `xl` 各种规格尺寸。因为这些类是从 `min-width: 0` 及以上开始引用的，所以不受媒体查询的约束，然而，其余的屏幕断点（设备解析）包含屏幕尺寸缩写。

对于 `xs` 屏幕，使用固定格式 `{property}{sides}-{size}` 命名 CSS 方法，对于 `sm`、`md`、`lg`、`xl` 使用 `{property}{sides}-{breakpoint}-{size}` 格式命名 CSS 方法。

如果 属性 是下列之一：

- `m` - 这个 Class 属性会设定 `margin` 值
- `p` - 这个 Class 属性会设定 `padding` 值

边缘设定:

- **t** - 这个 Class 属性会设定 **margin-top** 或 **padding-top**
- **b** - 这个 Class 属性会设定 **margin-bottom** 或 **padding-bottom**
- **l** - 这个 Class 属性会设定 **margin-left** 或 **padding-left**
- **r** - 这个 Class 属性会设定 **margin-right** 或 **padding-right**
- **x** - 这个 Class 属性会设定 ***-left** 和 ***-right** 两个值。
- **y** - 这个 Class 属性会设定 ***-top** 和 ***-bottom** 两个值
- 空白 - 这个 Class 属性会设定 **margin** 或 **padding** 元素的四个边。

尺寸规格定义:

- **0** - 这个 Class 属性会设定 **margin** 或 **padding** 的样式值为 **0**
- **1** - (默认时)这个 Class 属性会设定 **margin** 或 **padding** 以 **\$spacer * .25** 规格呈现
- **2** - (默认时) 这个 Class 属性会设定 **margin** 或 **padding** 以 **\$spacer * .5** 规格呈现
- **3** - (默认时)这个 Class 属性会设定 **margin** 或 **padding** 以 **\$spacer** 规格呈现
- **4** - (默认时) 这个 Class 属性会设定 **margin** 或 **padding** 以 **\$spacer * 1.5** 规格呈现
- **5** - (默认时)这个 Class 属性会设定 **margin** 或 **padding** 以 **\$spacer * 3** 规格呈现
- **auto** - 这个 Class 属性会设定 **margin** 值 **auto** (按浏览器默认值自由展现)。

(你也可以对**\$spacers** 的 Sass map 调整, 包括添加条目来增加更多尺寸。)

示例

以下是这些 Class 样式的代表性的示例:

```
.mt-0 {  
  margin-top: 0 !important;  
}  
  
.ml-1 {  
  margin-left: ($spacer * .25) !important;  
}  
  
.px-2 {  
  padding-left: ($spacer * .5) !important;  
  padding-right: ($spacer * .5) !important;  
}  
  
.p-3 {  
  padding: $spacer !important;  
}
```

水平居中

此外,Bootstrap 也包括一个 **.mx-auto** class 样式, 用于固定宽度的盒模型水平居中, 具有 **display: block** 和 **width** 设置水平边距内容的 **auto** 居中。

```
<div class="mx-auto" style="width: 200px;">  
  Centered element  
</div>
```

显示或隐藏(能见度)处理

使用可见性实用工具集控制元素的可见性，而不需要修改显示。

使用通用样式的 `visibility` 元属，这不会改变元素的 `display` 值，并且有助于大部分使用者隐藏内容，但仍然保留在屏幕阅读器中。

根据需要选用 `.visible` 或 `.invisible` 两个 Class 样式。

```
<div class="visible">...</div>
<div class="invisible">...</div>
```

```
// Class .visible {
  visibility: visible;
}
.invisible {
  visibility: hidden;
}
```

```
// Usage as a mixin
.element {
  @include invisible(visible);
}
.element {
  @include invisible(hidden);
}
```