### **Bootstrap 简介**

### **什么是Bootstrap？**

Bootstrap 是一个用于快速开发 Web 应用程序和网站的前端框架。Bootstrap 是基于 HTML、CSS、JAVASCRIPT 的。

http://v3.bootcss.com/

http://www.bootcss.com/

### **历史**

Bootstrap 是由 Twitter 的 Mark Otto 和 Jacob Thornton 开发的。Bootstrap 是 2011 年八月在 GitHub 上发布的开源产品。

### **为什么使用 Bootstrap？**

* 容易上手：只要您具备 HTML 和 CSS 的基础知识，您就可以开始学习 Bootstrap。
* 响应式设计：Bootstrap 的响应式 CSS 能够自适应于台式机、平板电脑和手机。更多有关响应式设计的内容详见 Bootstrap 响应式设计。
* 它为开发人员创建接口提供了一个简洁统一的解决方案。
* 它包含了功能强大的内置组件，易于定制。
* 它还提供了基于 Web 的定制。
* 它是开源的。

### **Bootstrap 包的内容**

* 基本结构：Bootstrap 提供了一个带有网格系统、链接样式、背景的基本结构。这将在 Bootstrap 基本结构 部分详细讲解。
* CSS：Bootstrap 自带以下特性：全局的 CSS 设置、定义基本的 HTML 元素样式、可扩展的 class，以及一个先进的网格系统。这将在Bootstrap CSS 部分详细讲解。
* 组件：Bootstrap 包含了十几个可重用的组件，用于创建图像、下拉菜单、导航、警告框、弹出框等等。这将在 布局组件 部分详细讲解。
* JavaScript 插件：Bootstrap 包含了十几个自定义的 jQuery 插件。您可以直接包含所有的插件，也可以逐个包含这些插件。这将在Bootstrap 插件 部分详细讲解。
* 定制：您可以定制 Bootstrap 的组件、LESS 变量和 jQuery 插件来得到您自己的版本。

### **下载 Bootstrap**

您可以从 <http://v3.bootcss.com/> 上下载 Bootstrap 的最新版本。当您点击这个链接时，您将看到如下所示的网页：



您会看到一个下载按钮：

* Download Bootstrap：下载 Bootstrap。点击该按钮，您可以下载 Bootstrap CSS、JavaScript 和字体的预编译的压缩版本。不包含文档和最初的源代码文件。

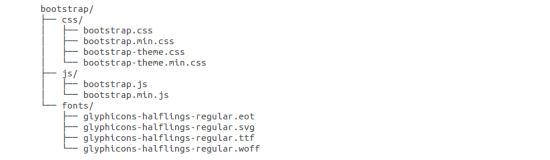
为了更好的了解和更方便的使用，我们将在本教程中使用 Bootstrap 的预编译版本。

由于文件是被编译过和压缩过的，在独立的功能开发中，您不必每次都包含这些独立的文件。

本教程编写时，使用的是最新版（Bootstrap 3）。

### **预编译的 Bootstrap**

当您下载了 Bootstrap 的已编译的版本，解压缩 ZIP 文件，您将看到下面的文件/目录结构：



如上图所示，可以看到已编译的 CSS 和 JS（bootstrap.），以及已编译压缩的 CSS 和 JS（bootstrap.min.）。同时也包含了 Glyphicons 的字体，这是一个可选的 Bootstrap 主题。

### **Bootstrap 源代码**

如果您下载了 Bootstrap 源代码，那么文件结构将如下所示：



* less、js 和 fonts 下的文件分别是 Bootstrap CSS、JS 和图标字体的源代码。
* dist 文件夹包含了上面预编译下载部分中所列的文件和文件夹。
* docs-assets、examples 和所有的 \*.html 文件是 Bootstrap 文档。

### **HTML 模板**

一个使用了 Bootstrap 的基本的 HTML 模板如下所示：

<!DOCTYPE html><html> <head> <title>Bootstrap 模板</title> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <!-- 引入 Bootstrap --> <link href="http://apps.bdimg.com/libs/bootstrap/3.3.0/css/bootstrap.min.css" rel="stylesheet"> <!-- HTML5 Shim 和 Respond.js 用于让 IE8 支持 HTML5元素和媒体查询 --> <!-- 注意： 如果通过 file:// 引入 Respond.js 文件，则该文件无法起效果 --> <!--[if lt IE 9]> <script src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script> <script src="https://oss.maxcdn.com/libs/respond.js/1.3.0/respond.min.js"></script> <![endif]--> </head> <body> <h1>Hello, world!</h1> <!-- jQuery (Bootstrap 的 JavaScript 插件需要引入 jQuery) --> <script src="https://code.jquery.com/jquery.js"></script> <!-- 包括所有已编译的插件 --> <script src="js/bootstrap.min.js"></script> </body></html>

在这里，您可以看到包含了 jquery.js、bootstrap.min.js 和bootstrap.min.css 文件，用于让一个常规的 HTML 文件变为使用了 Bootstrap 的模板。

有关上面代码段中每个元素的细节将在 Bootstrap CSS 概览 章节详细讲解。

### **Bootstrap CDN推荐**

Bootstrap 中文网 为 Bootstrap 专门构建了自己的免费 CDN 加速服务。基于国内云厂商的 CDN 服务，访问速度更快、加速效果更明显、没有速度和带宽限制、永久免费。Bootstrap 中文网还对大量的前端开源工具库提供了 CDN 加速服务，请进入BootCDN 主页查看更多可用的工具库。

<!-- 新 Bootstrap 核心 CSS 文件 --><link rel="stylesheet" href="http://cdn.bootcss.com/bootstrap/3.3.4/css/bootstrap.min.css"> <!-- 可选的Bootstrap主题文件（一般不用引入） --><link rel="stylesheet" href="http://cdn.bootcss.com/bootstrap/3.3.4/css/bootstrap-theme.min.css"> <!-- jQuery文件。务必在bootstrap.min.js 之前引入 --><script src="http://cdn.bootcss.com/jquery/1.11.2/jquery.min.js"></script> <!-- 最新的 Bootstrap 核心 JavaScript 文件 --><script src="http://cdn.bootcss.com/bootstrap/3.3.4/js/bootstrap.min.js"></script>

### **HTML 5 文档类型**

Bootstrap 使用了一些 HTML5 元素和 CSS 属性。为了让这些正常工作，您需要使用 HTML5 文档类型（Doctype）。 因此，请在使用 Bootstrap 项目的开头包含下面的代码段。

<!DOCTYPE html><html>....</html>

如果在 Bootstrap 创建的网页开头不使用 HTML5 的文档类型（Doctype），您可能会面临一些浏览器显示不一致的问题，甚至可能面临一些特定情境下的不一致，以致于您的代码不能通过 W3C 标准的验证。

### **移动设备优先**

移动设备优先是 Bootstrap 3 的最显著的变化。

在之前的 Bootstrap 版本中（直到 2.x），您需要手动引用另一个 CSS，才能让整个项目友好的支持移动设备。

现在不一样了，Bootstrap 3 默认的 CSS 本身就对移动设备友好支持。

Bootstrap 3 的设计目标是移动设备优先，然后才是桌面设备。这实际上是一个非常及时的转变，因为现在越来越多的用户使用移动设备。

为了让 Bootstrap 开发的网站对移动设备友好，确保适当的绘制和触屏缩放，需要在网页的 head 之中添加 viewport meta 标签，如下所示：

<meta name="viewport" content="width=device-width, initial-scale=1.0">

width 属性控制设备的宽度。假设您的网站将被带有不同屏幕分辨率的设备浏览，那么将它设置为 device-width 可以确保它能正确呈现在不同设备上。

initial-scale=1.0 确保网页加载时，以 1:1 的比例呈现，不会有任何的缩放。

在移动设备浏览器上，通过为 **viewport meta** 标签添加 user-scalable=no可以禁用其缩放（zooming）功能。

通常情况下，maximum-scale=1.0 与 user-scalable=no 一起使用。这样禁用缩放功能后，用户只能滚动屏幕，就能让您的网站看上去更像原生应用的感觉。

注意，这种方式我们并不推荐所有网站使用，还是要看您自己的情况而定！

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">

### **响应式图像**

<img src="..." class="img-responsive" alt="响应式图像">

通过添加 img-responsive class 可以让 Bootstrap 3 中的图像对响应式布局的支持更友好。

接下来让我们看下这个 class 包含了哪些 css 属性。

在下面的代码中，可以看到img-responsive class 为图像赋予了 max-width: 100%; 和 height: auto; 属性，可以让图像按比例缩放，不超过其父元素的尺寸。

.img-responsive { display: inline-block; height: auto; max-width: 100%;}

这表明相关的图像呈现为 inline-block。当您把元素的 display 属性设置为 inline-block，元素相对于它周围的内容以内联形式呈现，但与内联不同的是，这种情况下我们可以设置宽度和高度。

设置 height:auto，相关元素的高度取决于浏览器。

设置 max-width 为 100% 会重写任何通过 width 属性指定的宽度。这让图片对响应式布局的支持更友好。

### **排版与链接**

Bootstrap 排版、链接样式设置了基本的全局样式。分别是：

* 为 body 元素设置 background-color: #fff;
* 使用 @font-family-base、@font-size-base 和 @line-height-base a变量作为排版的基本参数
* 为所有链接设置了基本颜色 @link-color ，并且当链接处于 :hover状态时才添加下划线

这些样式都能在 scaffolding.less 文件中找到对应的源码。

### **Normalize.css**

Bootstrap 使用 [Normalize](http://necolas.github.io/normalize.css/) 来建立跨浏览器的一致性。

Normalize.css 是一个很小的 CSS 文件，在 HTML 元素的默认样式中提供了更好的跨浏览器一致性。

### **布局容器**

Bootstrap 需要为页面内容和栅格系统包裹一个 .container 容器。我们提供了两个作此用处的类。注意，由于 padding 等属性的原因，这两种 容器类不能互相嵌套。

.container 类用于固定宽度并支持响应式布局的容器。

<div class="container"> ...</div>

.container-fluid 类用于 100% 宽度，占据全部视口（viewport）的容器。

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

### **浏览器/设备支持**

Bootstrap 可以在最新的桌面系统和移动端浏览器中很好的工作。

旧的浏览器可能无法很好的支持。

下表为 Bootstrap 支持最新版本的浏览器和平台：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Chrome | Firefox | IE | Opera | Safari |
| **Android** | YES | YES | 不适用 | NO | 不适用 |
| **iOS** | YES | 不适用 | 不适用 | NO | YES |
| **Mac OS X** | YES | YES | 不适用 | YES | YES |
| **Windows** | YES | YES | YES\* | YES | NO |

# Bootstrap网格系统

### **简介**

栅格系统用于通过一系列的行（row）与列（column）的组合来创建页面布局，你的内容就可以放入这些创建好的布局中。下面就介绍一下 Bootstrap 栅格系统的工作原理：

* “行（row）”必须包含在 .container （固定宽度）或 .container-fluid （100% 宽度）中，以便为其赋予合适的排列（aligment）和内补（padding）。
* 通过“行（row）”在水平方向创建一组“列（column）”。
* 你的内容应当放置于“列（column）”内，并且，只有“列（column）”可以作为行（row）”的直接子元素。
* 类似 .row 和 .col-xs-4 这种预定义的类，可以用来快速创建栅格布局。Bootstrap 源码中定义的 mixin 也可以用来创建语义化的布局。
* 通过为“列（column）”设置 padding 属性，从而创建列与列之间的间隔（gutter）。通过为 .row 元素设置负值 margin 从而抵消掉为.container 元素设置的 padding，也就间接为“行（row）”所包含的“列（column）”抵消掉了padding。
* 负值的 margin就是下面的示例为什么是向外突出的原因。在栅格列中的内容排成一行。
* 栅格系统中的列是通过指定1到12的值来表示其跨越的范围。例如，三个等宽的列可以使用三个 .col-xs-4 来创建。
* 如果一“行（row）”中包含了的“列（column）”大于 12，多余的“列（column）”所在的元素将被作为一个整体另起一行排列。
* 栅格类适用于与屏幕宽度大于或等于分界点大小的设备 ， 并且针对小屏幕设备覆盖栅格类。 因此，在元素上应用任何 .col-md-<em> 栅格类适用于与屏幕宽度大于或等于分界点大小的设备 ， 并且针对小屏幕设备覆盖栅格类。 因此，在元素上应用任何 .col-lg- 不存在， 也影响大屏幕设备。

### **媒体查询**

在栅格系统中，我们在 Less 文件中使用以下媒体查询（media query）来创建关键的分界点阈值。

/ 超小屏幕（手机，小于 768px） /

/ 没有任何媒体查询相关的代码，因为这在 Bootstrap 中是默认的（还记得 Bootstrap 是移动设备优先的吗？） /

/ 小屏幕（平板，大于等于 768px） /

@media (min-width: @screen-sm-min) { ... }

/ 中等屏幕（桌面显示器，大于等于 992px） /

@media (min-width: @screen-md-min) { ... }

/ 大屏幕（大桌面显示器，大于等于 1200px） /

@media (min-width: @screen-lg-min) { ... }

我们偶尔也会在媒体查询代码中包含 max-width 从而将 CSS 的影响限制在更小范围的屏幕大小之内。

@media (max-width: @screen-xs-max) { ... }

@media (min-width: @screen-sm-min) and (max-width: @screen-sm-max) { ... }

@media (min-width: @screen-md-min) and (max-width: @screen-md-max) { ... }

@media (min-width: @screen-lg-min) { ... }

### **栅格参数**

通过下表可以详细查看 Bootstrap 的栅格系统是如何在多种屏幕设备上工作的。

|  | 超小屏幕 手机 (<768px) | 小屏幕 平板 (≥768px) | 中等屏幕 桌面显示器 (≥992px) | 大屏幕 大桌面显示器 (≥1200px) |
| --- | --- | --- | --- | --- |
| 栅格系统行为 | 总是水平排列 | 开始是堆叠在一起的，当大于这些阈值时将变为水平排列C | | |
| .container最大宽度 | None （自动） | 750px | 970px | 1170px |
| 类前缀 | .col-xs- | .col-sm- | .col-md- | .col-lg- |
| 列（column）数 | 12 | | | |
| 最大列（column）宽 | 自动 | ~62px | ~81px | ~97px |
| 槽（gutter）宽 | 30px （每列左右均有 15px） | | | |
| 可嵌套 | 是 | | | |
| 偏移（Offsets） | 是 | | | |
| 列排序 | 是 | | | |

### **从堆叠到水平排列**

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

<div class="row"> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div> <div class="col-md-1 myTest">.col-md-1</div></div><div class="row"> <div class="col-md-8 myTest">.col-md-8</div> <div class="col-md-4 myTest">.col-md-4</div></div><div class="row"> <div class="col-md-4 myTest">.col-md-4</div> <div class="col-md-4 myTest">.col-md-4</div> <div class="col-md-4 myTest">.col-md-4</div></div><div class="row"> <div class="col-md-6 myTest">.col-md-6</div> <div class="col-md-6 myTest">.col-md-6</div></div>

.col-md-1

.col-md-6

.col-md-6

.col-md-4

.col-md-4

.col-md-4

.col-md-4

.col-md-8

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

.col-md-1

### **流式布局容器**

将最外面的布局元素 .container 修改为 .container-fluid，就可以将固定宽度的栅格布局转换为 100% 宽度的布局。

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

### **移动设备和桌面屏幕**

是否不希望在小屏幕设备上所有列都堆叠在一起？那就使用针对超小屏幕和中等屏幕设备所定义的类吧，即 .col-xs-<em> 和 .col-md-。请看下面的实例，研究一下这些是如何工作的。

<!-- Stack the columns on mobile by making one full-width and the other half-width --><div class="row"> <div class="col-xs-12 col-md-8 myTest">.col-xs-12 .col-md-8</div> <div class="col-xs-6 col-md-4 myTest">.col-xs-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-xs-6 col-md-4 myTest">.col-xs-6 .col-md-4</div> <div class="col-xs-6 col-md-4 myTest">.col-xs-6 .col-md-4</div> <div class="col-xs-6 col-md-4 myTest">.col-xs-6 .col-md-4</div></div><!-- Columns are always 50% wide, on mobile and desktop --><div class="row"> <div class="col-xs-6 myTest">.col-xs-6</div> <div class="col-xs-6 myTest">.col-xs-6</div></div>

.col-xs-12 .col-md-8

.col-xs-6 .col-md-4

.col-xs-6 .col-md-4

.col-xs-6 .col-md-4

.col-xs-6 .col-md-4

.col-xs-6

.col-xs-6

### **手机、平板、桌面**

在上面案例的基础上，通过使用针对平板设备的 .col-sm-\* 类，我们来创建更加动态和强大的布局吧。

<div class="row"> <div class="col-xs-12 col-sm-6 col-md-8">.col-xs-12 .col-sm-6 .col-md-8</div> <div class="col-xs-6 col-md-4">.col-xs-6 .col-md-4</div></div><div class="row"> <div class="col-xs-6 col-sm-4">.col-xs-6 .col-sm-4</div> <div class="col-xs-6 col-sm-4">.col-xs-6 .col-sm-4</div> <!-- Optional: clear the XS cols if their content doesn't match in height --> <div class="clearfix visible-xs-block"></div> <div class="col-xs-6 col-sm-4">.col-xs-6 .col-sm-4</div></div>

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

.col-xs-6 .col-md-4

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

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

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

### **多余的列（column）将另起一行排列**

如果在一个 .row 内包含的列（column）大于12个，包含多余列（column）的元素将作为一个整体单元被另起一行排列。

<div class="row"> <div class="col-xs-9">.col-xs-9</div> <div class="col-xs-4">.col-xs-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-xs-6">.col-xs-6<br>Subsequent columns continue along the new line.</div></div>

.col-xs-9

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

.col-xs-6  
Subsequent columns continue along the new line.

### **响应式列重置**

即便有上面给出的四组栅格class，你也不免会碰到一些问题，例如，在某些阈值时，某些列可能会出现比别的列高的情况。为了克服这一问题，建议联合使用 .clearfix 和 响应式工具类。

<div class="row"> <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div> <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div>  <!-- Add the extra clearfix for only the required viewport --> <div class="clearfix visible-xs-block"></div> <p> <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div> <div class="col-xs-6 col-sm-3">.col-xs-6 .col-sm-3</div></div>

.col-xs-6 .col-sm-3   
Resize your viewport or check it out on your phone for an example.

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

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

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