**什么是 Bootstrap 网格系统（Grid System）？**

Bootstrap 官方文档中有关网格系统的描述：

http://www.runoob.com/images/quote.pngBootstrap 包含了一个响应式的、移动设备优先的、不固定的网格系统，可以随着设备或视口大小的增加而适当地扩展到 12 列。它包含了用于简单的布局选项的预定义类，也包含了用于生成更多语义布局的功能强大的混合类。

让我们来理解一下上面的语句。Bootstrap 3 是移动设备优先的，在这个意义上，Bootstrap 代码从小屏幕设备（比如移动设备、平板电脑）开始，然后扩展到大屏幕设备（比如笔记本电脑、台式电脑）上的组件和网格。

### 移动设备优先策略

* **内容**
  + 决定什么是最重要的。
* **布局**
  + 优先设计更小的宽度。
  + 基础的 CSS 是移动设备优先，媒体查询是针对于平板电脑、台式电脑。
* **渐进增强**
  + 随着屏幕大小的增加而添加元素。

响应式网格系统随着屏幕或视口（viewport）尺寸的增加，系统会自动分为最多12列。

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 4 | | | | 4 | | | | 4 | | | |
| 4 | | | | 8 | | | | | | | |
| 6 | | | | | | 6 | | | | | |
| 12 | | | | | | | | | | | |

**Bootstrap 网格系统（Grid System）的工作原理**

网格系统通过一系列包含内容的行和列来创建页面布局。下面列出了 Bootstrap 网格系统是如何工作的：

* 行必须放置在 **.container** class 内，以便获得适当的对齐（alignment）和内边距（padding）。
* 使用行来创建列的水平组。
* 内容应该放置在列内，且唯有列可以是行的直接子元素。
* 预定义的网格类，比如 **.row** 和 **.col-xs-4**，可用于快速创建网格布局。LESS 混合类可用于更多语义布局。
* 列通过内边距（padding）来创建列内容之间的间隙。该内边距是通过 **.rows** 上的外边距（margin）取负，表示第一列和最后一列的行偏移。
* 网格系统是通过指定您想要横跨的十二个可用的列来创建的。例如，要创建三个相等的列，则使用三个 **.col-xs-4**。

**媒体查询**

媒体查询是非常别致的"有条件的 CSS 规则"。它只适用于一些基于某些规定条件的 CSS。如果满足那些条件，则应用相应的样式。

Bootstrap 中的媒体查询允许您基于视口大小移动、显示并隐藏内容。下面的媒体查询在 LESS 文件中使用，用来创建 Bootstrap 网格系统中的关键的分界点阈值。

/\* 超小设备（手机，小于 768px） \*/

/\* Bootstrap 中默认情况下没有媒体查询 \*/

/\* 小型设备（平板电脑，768px 起） \*/

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

/\* 中型设备（台式电脑，992px 起） \*/

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

/\* 大型设备（大台式电脑，1200px 起） \*/

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

我们有时候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) { ... }

媒体查询有两个部分，先是一个设备规范，然后是一个大小规则。在上面的案例中，设置了下列的规则：

让我们来看下面这行代码：

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

对于所有带有 *min-width: @screen-sm-min* 的设备，如果屏幕的宽度小于 *@screen-sm-max*，则会进行一些处理。

Bootstrap 网格系统实例：堆叠的水平

让我们来看一个带有简单布局的网格实例：包含两个列，每个列包含两个段落。（在这里，为每个列分别定义了样式，您可以避免这么做。）

实例

<div class="container">

<h1>Hello, world!</h1>

<div class="row">

<div class="col-md-6" style="background-color: #dedef8; box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">

<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do

</p>

<p>Sed ut perspiciatis unde omnis iste natus error sit voluptatem

</p>

</div>

<div class="col-md-6" style="background-color: #dedef8;box-shadow: inset 1px -1px 1px #444, inset -1px 1px 1px #444;">

<p>Sed ut perspiciatis unde omnis iste natus error sit voluptatem

</p>

<p> Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet,

</p>

</div>

</div>

</div>

* **<div class="container">...</div>** 元素被添加，确保居中和最大宽度。
* 一旦添加了容器，接下来您需要考虑以行为单位。添加 **<div class="row">...</div>**，并在行内添加列 **<div class="col-md-6"></div>**。
* 网格中的每一行是由 12 个单元组成的，您可以使用这些单元定义列的尺寸。在我们的实例中，有两个列，每个列由 6 个单元组成，即 6+6=12。

您可以尝试其他更多的选项，比如 **<div class="col-md-3"></div>** 和 **<div class="col-md-9"></div>** 或 **<div class="col-md-7"></div>** 和 **<div class="col-md-5"></div>**。

您可以尝试一下，但要确保总和总是 12

我们已经看过 [中型和大型设备](http://www.runoob.com/bootstrap/bootstrap-grid-system-example2.html) 。现在，让我们一起看另一个实例，我们将让它同样适用于小型手机。我们要把平板电脑的列分割为 25%/75%，我们将添加如下选项：

<div class="col-sm-3 col-md-6 col-lg-4">....</div>

<div class="col-sm-9 col-md-6 col-lg-8">....</div>

受限于公司屏幕尺寸，并不能试出大型设备的网页效果