目录

[1.基本的HTML标签 1](#_Toc3574)

[1. 1布局标签 1](#_Toc28087)

[1.2文本标签 1](#_Toc20986)

[1.3序列化标签 1](#_Toc31049)

[1.4表格标签 2](#_Toc11013)

[1.5表单标签 2](#_Toc29450)

[2. 基本的CSS 列举/布局: 4](#_Toc18109)

[2.1css中的选择器 4](#_Toc15886)

[2. 2css 样式 4](#_Toc24522)

[3.文本相关 6](#_Toc1722)

[3.1文本标签 6](#_Toc17393)

[3.2字体样式 6](#_Toc13030)

[3.3引用字体 14](#_Toc31787)

[4.图形相关 14](#_Toc36)

[4.1渐变 14](#_Toc11857)

[4.2阴影 14](#_Toc10543)

[4.3形状(裁剪) 14](#_Toc19799)

[4.4形变 14](#_Toc29640)

[4.5平移 15](#_Toc5162)

[4.6旋转 15](#_Toc21136)

[4.7缩放 15](#_Toc26795)

[4.8透明度 15](#_Toc25248)

[滤镜 15](#_Toc28514)

[4.9动画 15](#_Toc2795)

[4.10计时器 15](#_Toc7639)

[5.Canvas 16](#_Toc29617)

[5.1什么是canvas: 16](#_Toc18617)

[5.2 Canvas 16](#_Toc32396)

[6.视频播放器 18](#_Toc21564)

[6.1具体语法如下所示 18](#_Toc27615)

[6.2<video>标签常用属性 18](#_Toc8271)

[6.3Video对象的常用方法 19](#_Toc167)

[6.4Video对象的常用方法和常用属性 19](#_Toc22425)

[6.5Video对象的常用事件 20](#_Toc22533)

[7.音频 20](#_Toc16231)

[7.1语法 20](#_Toc9729)

[7.2< audio >标签的常用属性如下表所示。 21](#_Toc22503)

[7.3Audio对象的常用方法 21](#_Toc7493)

[7.4Audio对象的常用属性 22](#_Toc22378)

[7.5Audio对象的常用事件 22](#_Toc30701)

[8.图标库 24](#_Toc8321)

[8.1基本图标 24](#_Toc25501)

[8.2大图标 24](#_Toc4655)

[8.3固定宽度 24](#_Toc21370)

[8.4用于列表 24](#_Toc19080)

[8.5边框与对齐 25](#_Toc3771)

[8.6动画 25](#_Toc30585)

[8.7组合使用 25](#_Toc18667)

[9.动画库 26](#_Toc28294)

[9.1 Animate.css动画库的简介 26](#_Toc14611)

[9.2 Animate.css动画库的使用 26](#_Toc19393)

[10.响应式----媒体查询、栅格系统 28](#_Toc3943)

[10.1媒体查询(@media 查询) 28](#_Toc6481)

[11.栅格系统 28](#_Toc25902)

[11.1什么是 Bootstrap 网格系统（Grid System）？ 28](#_Toc19510)

[11.2网格选项 29](#_Toc24818)

[11.3基本的网格结构:container→row→col 29](#_Toc2099)

[11.4偏移列 30](#_Toc22853)

[11.5嵌套列 30](#_Toc27651)

[11.6列排序 30](#_Toc21148)

[12.Flex布局 30](#_Toc28743)

[12.1什么是Flex? 30](#_Toc22798)

[12.2基本概念 30](#_Toc30416)

[12.3容器的属性 31](#_Toc31992)

[12.4项目的属性 32](#_Toc21798)

[13.bootstrap 32](#_Toc12758)

[13.1安装和依赖 32](#_Toc15499)

[13.2使用 33](#_Toc2256)

[13.3网格系统 33](#_Toc12995)

[13.3.1 列偏移 34](#_Toc19664)

[13.3.2 嵌套列 34](#_Toc10856)

[13.3.3列排序 34](#_Toc29719)

[13.4组件库 34](#_Toc32404)

[13.4.1使用方法 35](#_Toc15277)

[13.5主题切换 35](#_Toc29990)

# 1.基本的HTML标签

http://css.cuishifeng.cn/ css样式表

## 1布局标签

|  |  |
| --- | --- |
| <div>标签 | 定义文档中的分区或节，可以把文档分割为独立的、不同的部分，主要用于布局。 |
| <aside>标签 | 可用作文章的侧栏,跟article相关的边角内容，或辅助信息 |
| <header>标签 | 定义页面的头部（介绍信息） |
| <sectio>标签 | 定义文档中的节（section、区段）。比如章节、页眉、页脚或文档中的其他部分 |
| <footer> 标签 | 定义文档或节的页脚，通常包含文档的作者、版权信息、使用条款链接、联系信息等等 |
| <article>标签 | 规定文章独立的其他内容，比如：标题、内容、评论 |
| <hgroup> 标签 | 标题组合 |
| <figure>标签 | 独立的流内容 |
| <figcaption>标签 | figure的标题 |
| <nav>标签 | 导航相关 |

## 1.2文本标签

|  |  |
| --- | --- |
| h1-h6标签 | 定义标题 |
| p标签 | 定义段落 |
| b/strong标签 | 加粗 |
| em标签 | 表示强调的文本，斜体 |
| strong标签 | 表示重要文本 |
| u标签 | 下划线 |
| s标签 | 删除线 |
| br标签 | 表示回车换行 |
| hr标签 | 表示水平线 |
| span标签 | 用来组合文档中的行内元素 |
| pre标签 | 定义预格式化的文本，保持原有格式的一种标签。 |

## 1.3序列化标签

ul和li无序列表标签:

<ul>

<li>1111</li>

<li>2222</li>

<li>333</li>

</ul>

ol和li有序列表标签:

<ol>

<li>1111</li>

<li>2222</li>

<li>333</li>

</ol>

dl标签定义了定义列表，dl标签用于结合 dt（定义列表中的项目）和 dd（描述列表中的项目）:

<dl>

<dt>1111</dt>

<dd>2222</dd>

</dl>

## 1.4表格标签

table标签和tr标签，th标签和td标签，合并单元格:

<table width="100%" height="193" border="1" cellpadding="0" cellspacing="0" bordercolor="#FF0000" background="">

<tr>

<th>标题</th>

<th>标题</th>

</tr>

<tr>

<!-- 合并横向单元格 -->

<td colspan="2" nowrap="nowrap">&nbsp;</td>

</tr>

<tr>

<td></td>

<!-- 合并纵向单元格 -->

<td rowspan="2"> </td>

</tr>

<tr>

<td height="16"> </td>

</tr>

</table>

## 1.5表单标签

form标签:

定义提交方式、提交地址、表单字符集以及如何对其进行编码，需要提交的表单一定要放在form标签内。

<form action="https://www.baidu.com/s" method="get" autocomplete="no" name="form01">

<p>

密码框:<input type="password" name="密码框" value="666666"/>

</p>

<p>

性别:(单选)

<br>

<!-- checked="checked" 默认选择 -->

<!-- 单选框 radio -->

男：<input type="radio" name="性别" value="男" checked="checked" />

&nbsp;

女：<input type="radio" name="性别" value="女" />

</p>

<p>

兴趣爱好：checkbox(多选)

唱<input type="checkbox" name="复选框" value="唱" />

跳<input type="checkbox" name="复选框" value="跳" />

rap<input type="checkbox" name="复选框" value="rap" />

篮球<input type="checkbox" name="复选框" value="篮球" />

</p>

<p>

复选框:<input type="checkbox" name="" value="复选框" />

</p>

<p>

email:<input type="email" name="邮件" value="" />

</p>

<p>

滑轮range: <input type="range" name="" id="" value="range" />

</p>

<p>

隐藏域：<input type="hidden" name="隐藏域" id="" value="隐藏域" />

</p>

<p>

选择文件：<input type="file" name="file" id="" value="" />

</p>

多行文本输入控件textarea:<textarea name="textarea">type的其他类型： url, email, number, image, file, hidden,range

</textarea>

<p>

百度一下:<input type="text" name="wd" value="MCB" />

</p>

<p>

提交:<input type="submit" value="提交" />

重置:<input type="reset" value="重置" />

</p>

</form>

textarea标签，设置文本区内的可见行数和宽度:

<textarea name="content" cols="30" rows="10">大段文本输入框</textarea>

select标签和option标签下拉列表

单选菜单列表框:

<select name="user">

<option value="1">123</option>

<option value="2" selected="selected">345</option>

</select>

多选列表下拉框，shift加鼠标单击，可以连续选择多个选择，CTRL+鼠标点击，可以点击多个。

<select name="user" size="10" multiple="multiple">

<option value="1">大哥</option>

<option value="2" selected="selected">谷歌</option>

<option value="3">胡歌</option>

</select>

# 基本的CSS 列举/布局:

## 2.1css中的选择器

选择器 - 用来定位标签:

|  |  |  |
| --- | --- | --- |
| 选择器 |  | 描述 |
| 通用选择器 | \*{ } | 选择所有元素 |
| 标签选择器 | a{} / div{ } | 根据类名选择 |
| 类选择器 | .className{ } | 选择所有该标签 |
| id选择器 | #id{ } | 根据id名选择 |
| 属性选择器 | [attr=''] | 属性符合的元素 |
| 并集合(选择器拼凑) | div, .className | 条件拼凑 |
| 后代选择器 | .className li{ } | 选择后代所有li |
| 子代选择器 | .className>li | 仅限于第一层，子元素 |
| 兄弟选择器 | p+a{} | 跟p同级别紧跟着的a |
| 伪类选择器 | p::first{ } | 第一个p标签 |

## 2css 样式

**css背景:**

CSS 背景属性用于定义HTML元素的背景。

CSS 属性定义背景效果:

background-color定义了元素的背景颜色

body { background-color: #b0c4de }

background-image描述了元素的背景图像

body { background-image: url(‘1.jpg’) }

background-repeat让背景图像平铺或者不平铺

body { background-image: url(‘1.jpg’) ; background-repeat: no-repeat }

background-position背景图像- 设置定位

body

{

background-image:url('img\_tree.png');

background-repeat:no-repeat;

background-position:right top;

margin-right:200px;

}

边框属性Border

|  |  |
| --- | --- |
| border-style: | dotted;(点线) dashed;(虚线) solid; double;(双线) groove;(槽线) ridge;(脊状) inset;(凹陷) outset; |
| border-width: | 边框宽度 |
| border-color:#; | 边框颜色 |
| border：width style color; /\*简写\*/ | 简写方法 |
| border-radius | 四个值: 第一个值为左上角，第二个值为右上角，第三个值为右下角，第四个值为左下角。三个值: 第一个值为左上角, 第二个值为右上角和左下角，第三个值为右下角  两个值: 第一个值为左上角与右下角，第二个值为右上角与左下角  一个值： 四个圆角值相同 |

列表属性List-style

|  |  |
| --- | --- |
| 类型 | list-style-type: disc;(圆点) circle;(圆圈) square;(方块) decimal;(数字) lower-roman;(小罗码数字) |
| 位置 | list-style-position: outside;(外) inside; |
| 图像 | list-style-image: url(..); |

定位属性Position

|  |  |
| --- | --- |
| Position | absolute; relative; static;fixed;sticky |
| visibility | inherit; visible; hidden; |
| Overflow(溢出显示) | visible; hidden; scroll; auto; |
| Float(浮动) | Left;right;none;inherit |

Css 渐变

线性渐变（Linear Gradients）- 向下/向上/向左/向右/对角方向

#grad1 {

height: 200px;

background: -webkit-linear-gradient(left, red, blue); /\* Safari 5.1 - 6.0 \*/

background: -o-linear-gradient(right, red, blue); /\* Opera 11.1 - 12.0 \*/

background: -moz-linear-gradient(bottom, red, blue); /\* Firefox 3.6 - 15 \*/

background: linear-gradient(red, blue); /\* 标准的语法（必须放在最后） \*/

}

径向渐变（Radial Gradients）- 由它们的中心定义:

设置形状: shape 参数定义了形状。它可以是值 circle 或 ellipse。其中，circle 表示圆形，ellipse 表示椭圆形。默认值是 ellipse。

#grad1 {

height: 150px;

width: 200px;

background: -webkit-radial-gradient(red 5%, green 15%, blue 60%); /\* Safari 5.1 - 6.0 \*/

background: -o-radial-gradient(red 5%, green 15%, blue 60%); /\* Opera 11.6 - 12.0 \*/

background: -moz-radial-gradient(red 5%, green 15%, blue 60%); /\* Firefox 3.6 - 15 \*/

background: radial-gradient(red 5%, green 15%, blue 60%); /\* 标准的语法（必须放在最后） \*/

}

css按钮

.button2 {background-color: #008CBA;} /\* Blue \*/

.button3 {background-color: #f44336;} /\* Red \*/

.button4 {background-color: #e7e7e7; color: black;} /\* Gray \*/

.button5 {background-color: #555555;} /\* Black \*/

# 3.文本相关

参考 [http://css.cuishifeng.cn](http://css.cuishifeng.cn/)

[https://www.runoob.com](https://www.runoob.com/)

## 3.1文本标签

|  |  |
| --- | --- |
| 标签 | 描述 |
| 标题标签 | HTML中，定义了6级标题，分别为h1、h2、h3、h4、h5、h6，每级标题的字体大小依次递减，1级标题字号最大，6级标题字号最小 |
| <br>标签与<wbr>标签 | <p>标签用于定义段落。 |
| <br>标签与<wbr>标签 | <br>标签可插入一个简单的换行符，用来输入空行，而不是分隔段落。  <wbr>规定在文本中的何处适合添加换行符。作用是建议浏览器在这个标记处可以断行,只是建议而不是必定会在此处断行,还要根据整行文字长度而定。除了Internet Explorer，其他所有浏览器都支持<wbr> |
| <details>标签与<summary>标签 | <details>标签用于描述文档或文档某个部分的细节，目前只有Chrome浏览器支持<details>标签，可以与<summary>标签配合使用，<summary>标签用于定义这个描述文档的标题。 |
| <bdi>标签 | <bdi>标签用于设置一段文本，使其脱离其父标签的文本方向设置。 |
| <ruby>标签、<rt>标签与<rp>标签 | <ruby>标签用于定义ruby注释（中文注音或字符）。与<rt>标签一同使用。  <rt>标签用于定义字符(中文注音或字符)的解释或发音。  <rp>标签在rudy注释中使用,以定义不支持<rudy>标签的浏览器所显示的内容。 |
| <mark>标签 | <mark>标签主要用来在视觉上向用户呈现那些需要突出显示或高亮显示的文字，典型应用是搜索结果中高亮显示搜素关键字。 |
| <time>标签 | <time>标签用于定义日期或时间，也可以两者同时。 |
| <meter>标签 | <meter>标签用于定义度量衡，仅用于已知最大和最小值的度量。 |
| <progress>标签 | <progress>标签用于定义任何类型任务的运行进度，可以使用<progress>标签显示JavaScript中时间函数的进程。 |

## 3.2字体样式

|  |  |
| --- | --- |
| 属性 | 取值 |
| 文本字体样式font-style： | normal：指定文本字体样式为正常的字体  italic：指定文本字体样式为斜体。对于没有设计斜体的特殊字体，如果要使用斜体外观将应用oblique  oblique：指定文本字体样式为倾斜的字体。人为的使文字倾斜，而不是去选取字体中的斜体字 |
| 把段落设置为小型大写字母体font-variant： | normal：正常的字体  small-caps：小型的大写字母字体 |
| 文本字体的粗细font-weight： | normal：正常的字体。相当于数字值400  bold：粗体。相当于数字值700。  bolder：定义比继承值更重的值  lighter：定义比继承值更轻的值  用数字表示文本字体粗细。取值范围：100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 |
| 字体尺寸font-size： | (常用)用百分比指定文字大小。其百分比取值是基于父对象中字体的尺寸。不允许负值。 |
| 文本的字体名称font-family： | 字体名称。按优先顺序排列。以逗号隔开。如果字体名称包含空格或中文，则应使用引号括起 |
| 文字是否横向拉伸变形font-stretch： | normal：  正常文字宽度  ultra-condensed：  比正常文字宽度窄4个基数。  extra-condensed：  比正常文字宽度窄3个基数。  condensed：  比正常文字宽度窄2个基数。  semi-condensed：  比正常文字宽度窄1个基数。  semi-expanded：  比正常文字宽度宽1个基数。  expanded：  比正常文字宽度宽2个基数。  extra-expanded：  比正常文字宽度宽3个基数。  ultra-expanded：  比正常文字宽度宽4个基数。 |
| 文本的大小写text-transform： | none：  无转换  capitalize：  将每个单词的第一个字母转换成大写  uppercase：  将每个单词转换成大写  lowercase：  将每个单词转换成小写 |
| 空格的处理方式white-space： | normal：  默认处理方式。  pre：  用等宽字体显示预先格式化的文本，不合并文字间的空白距离，当文字超出边界时不换行。可查阅pre对象  nowrap：  强制在同一行内显示所有文本，合并文本间的多余空白，直到文本结束或者遭遇br对象。  pre-wrap：  用等宽字体显示预先格式化的文本，不合并文字间的空白距离，当文字碰到边界时发生换行。  pre-line：  保持文本的换行，不保留文字间的空白距离，当文字碰到边界时发生换行。 |
| 文本的字内换行行为word-break： | normal：  依照亚洲语言和非亚洲语言的文本规则，允许在字内换行。  keep-all：  与所有非亚洲语言的normal相同。对于中文，韩文，日文，不允许字断开。适合包含少量亚洲文本的非亚洲文本。  break-all：  该行为与亚洲语言的normal相同。也允许非亚洲语言文本行的任意字内断开。该值适合包含一些非亚洲文本的亚洲文本，比如使连续的英文字母间断行。 |
| 指定容器的边界时是否断行word-wrap： | normal：  允许内容顶开或溢出指定的容器边界。  break-word：  内容将在边界内换行。如果需要，单词内部允许断行。 |
| 指定容器的边界时是否断行overflow-wrap： | normal：  允许内容顶开或溢出指定的容器边界。  break-word：  内容将在边界内换行。如果需要，单词内部允许断行。 |
| 内容的水平对齐方式text-align： | left：  内容左对齐。  center：  内容居中对齐。  right：  内容右对齐。  justify：  内容两端对齐，但对于强制打断的行（被打断的这一行）及最后一行（包括仅有一行文本的情况，因为它既是第一行也是最后一行）不做处理。（CSS3）  start：  内容对齐开始边界。（CSS3）  end：  内容对齐结束边界。（CSS3）  match-parent：  这个值和 inherit 表现一致，只是该值继承的 start 或 end 关键字是针对父母的 <' [direction](http://css.cuishifeng.cn/direction.html) '> 值并计算的，计算值可以是 left 和 right 。（CSS3）  justify-all：  效果等同于 justify，但还会让最后一行也两端对齐。（CSS3） |
| 一个块内的最后一行或者被强制打断的行的对齐方式text-align-last： | auto：  无特殊对齐方式。  left：  内容左对齐。  center：  内容居中对齐。  right：  内容右对齐。  justify：  内容两端对齐。  start：  内容对齐开始边界。  end：  内容对齐结束边界。 |
| 调整文本使用的对齐方式text-justify： | auto：  允许浏览器用户代理确定使用的两端对齐法则。  none：  禁止两端对齐。  inter-word：  通过增加字之间的空格对齐文本。该行为是对齐所有文本行最快的方法，它的两端对齐行为对段落的最后一行无效。  inter-ideograph：  为表意字文本提供完全两端对齐，增加或减少表意字和词间的空格。  inter-cluster：  调整文本无词间空格的行。这种模式的调整是用于优化亚洲语言文档的  distribute：  通过增加或减少字或字母之间的空格对齐文本，适用于东亚文档，尤其是泰国。  kashida：  通过拉长选定点的字符调整文本。这种调整模式是特别为阿拉伯脚本语言提供的。需要IE5.5+支持 |
| 单词之间的最小，最大和最佳间隙word-spacing： | normal：  默认间隔  用长度值指定间隔。可以为负值。  用百分比指定间隔。可以为负值。（CSS3） |
| 字符之间的最小，最大和最佳间隙letter-spacing： | normal：  默认间隔  用长度值指定间隔。可以为负值。  用百分比指定间隔。可以为负值。（CSS3） |
| 文本的缩进text-indent： | 用长度值指定文本的缩进。可以为负值。  用百分比指定文本的缩进。可以为负值。  each-line：  定义缩进作用在块容器的第一行或者内部的每个强制换行的首行，软换行不受影响。（CSS3）  hanging：  反向所有被缩进作用的行。（CSS3） |
| 行高。即字体最底端与字体内部顶端之间的距离line-height： | normal：  允许内容顶开或溢出指定的容器边界。  用长度值指定行高。不允许负值  用百分比指定行高，其百分比取值是基于字体的高度尺寸。不允许负值 |
| 移动端页面中对象文本的大小调整text-size-adjust： | auto：  文本大小根据设备尺寸进行调整。  none：  文本大小不会根据设备尺寸进行调整。  用百分比来指定文本大小在设备尺寸不同的情况下如何调整 |
| 文本的装饰text-decoration：  (多用于去除<a>的默认样式) | 指定文本装饰的种类。相当于CSS2.1的 <' text-decoration '> 属性，可取值：none | underline | overline | line-through | blink {}  指定文本装饰的样式  指定文本装饰的颜色 |
| 文本装饰线条的位置text-decoration-line： | none：  指定文字无装饰  underline：  指定文字的装饰是下划线  overline：  指定文字的装饰是上划线  line-through：  指定文字的装饰是贯穿线  blink：  指定文字的装饰是闪烁。 |
| 文本装饰线条的颜色text-decoration-color： | 指定颜色 |
| 文本装饰线条的形状text-decoration-style： | solid：  实线  double：  双线  dotted：  点状线条  dashed：  虚线  wavy：  波浪线 |
| 设置对象中的下划线的位置text-underline-position： | auto：  用户代理可能会使用任意算法确定下划线的位置。  under：  下划线的定位与元素内容盒子的下边缘相关  left：  下划线的定位与元素内容盒子的左边缘相关  right：  下划线的定位与元素内容盒子的右边缘相关 |
| 文字阴影及模糊效果text-shadow： | 水平偏移值，垂直偏移值，阴影模糊值 |
| 文本流的方向direction： | ltr：  文本流从左到右。  rtl：  文本流从右到左。 |

## 3.3引用字体

@font-face {

font-family:;

src: url();

}

# 4.图形相关

## 4.1渐变

从上到下:(颜色后可加百分比)

background: linear-gradient(red, blue);

从左到右:

background: linear-gradient(to right, red , blue);

对角:

background: linear-gradient(to bottom right, red , blue);

使用透明度:

background:linear-gradient(toright,rgba(255,0,0,0), rgba(255,0,0,1));

径向渐变:(颜色后可加百分比)

background: radial-gradient(red, green, blue);

径向渐变可设置形状(椭圆):

background: radial-gradient(ellipse, red, yellow, green);

## 4.2阴影

滤镜// filter: drop-shadow(8px 8px 10px red);

## 4.3形状(裁剪)

圆 clip-path: circle(半径 at 圆心点);

椭圆 clip-path: ellipse(半径,半径 at 中心点);

多边形 clip-path: polygon(坐标,坐标,...);所有点依次连接形 成的闭合区域

## 4.4形变

skew()：

指定对象skew transformation（斜切扭曲）。第一个参数对应X轴，第二个参数对应Y轴。如果第二个参数未提供，则默认值为0

skewx()：

指定对象X轴的（水平方向）扭曲

skewy()：

指定对象Y轴的（垂直方向）扭曲

## 4.5平移

transform: translateX();X轴（水平方向）的平移

transform: translateY();Y轴（垂直方向）的平移

transform: translate(a,b);

## 4.6旋转

rotate()：

指定对象的2D rotation（2D旋转），需先有 <' transform-origin '> 属性的定义

<' transform-origin '> 设置变换基点

## 4.7缩放

scale()：

指定对象的2D scale（2D缩放）。第一个参数对应X轴，第二个参数对应Y轴。如果第二个参数未提供，则默认取第一个参数的值

scalex()：

指定对象X轴的（水平方向）缩放

scaley()：

指定对象Y轴的（垂直方向）缩放

## 4.8透明度

opacity: ;

属性值从0.0 - 1.0。值越小，使得元素更加透明

4.9滤镜

filter: grayscale(100%); //灰度

filter: blur(px);高斯迷糊

filter: brightness(%);亮度

filter: contrast(%);对比度

filter: invert(%);图像反转

filter: opacity(%);图像透明度

filter: sepia(%);褐度

## 4.10动画

animation-name: xxx;//动画名称

animation-duration: 5s;动画的持续时间

animation-timing-function: linear; //动画的过渡方式

animation-delay: 2s;//延迟

animation-iteration-count: infinite;//动画被播放的次数

animation-direction: alternate;//动画是否交替运行

@keyframes xxx{

from{...}

to{...}

}

## 4.11计时器

var myVar=setInterval(function(){} ,毫秒)

clearInterval(myVar) 关闭计时器

myVar=setTimeout(function(){} ,毫秒) 执行一次

clearTimeout(myVar);

要使用clearTimeout() 方法, 你必须在创建超时方法中（setTimeout）使用全局变量

如果函数还未被执行，你可以使用 clearTimeout() 方法来停止执行函数代码

# 5.Canvas

<https://www.runoob.com/html/html5-canvas.html>

<https://blog.csdn.net/weixin_42372054/article/details/82790677>

## 5.1什么是canvas:

HTML5 <canvas> 元素用于图形的绘制，通过脚本 (通常是JavaScript)来完成.<canvas> 标签只是图形容器，必须使用脚本来绘制图形。

## 5.2 Canvas

#### 5.2.1使用canvas绘制图形的基本步骤总结如下：

（1）创建画布：<canvas></canvas>

（2）准备画笔（创建 context 对象）：canvas.getContext('2d');

（3）开始路径规划 ：context.beginPath();

//(beginPath()作用是在绘制新路径时，使得新路径的绘制不会影响原有的路径)

（4）移动起始点 ：context.moveTo(x, y);

（5）绘制线(矩形、圆形、图片...) ：context.lineTo(x, y);

（6）闭合路径：context.closePath();

//(closePath()的意思是从路径绘制开始的点到绘制结束点，这两个连接起来，形成闭合路径)

绘制描边 ：context.stroke();

绘制完毕后就不能修改颜色等属性，所以绘制是最后一步,绘制图之前一定先 beginPath()

#### 5.2.2使用canvas绘制图像

绘制矩形

ctx.strokeRect(x, y, 矩形width，矩形height) -------绘制矩形（不填色）

ctx.fillRect(x, y, 矩形width，矩形height) ------- 绘制已填充矩形

clearRect(x, y, 矩形width，矩形height) 清除指定的矩形区域，然后这块区域会变的完全透明。

绘制圆弧 -----弧度：(Math.PI/180)\*360

方法一:

ctx.arc(x, y, r, 起始弧度startAngle， 终点弧度endAngle，anticlockwise)

ctx.stroke()//绘制描边

//以(x,y)为圆心,r为半径，从startAngle角度开始到endAngle角度结束,anticlockwise是布尔值,true(默认)表示逆时针,false表示是时针。

方法二:

ctx.arcTo(x1, y1, x2,y2,radius)

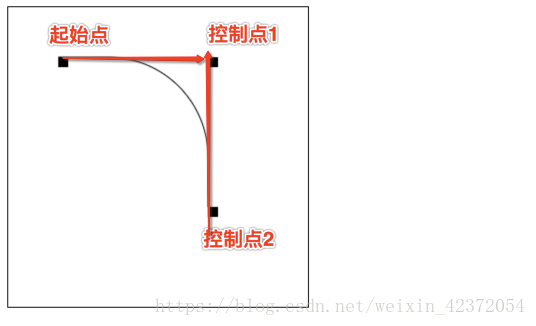
arcTo方法的说明：

​绘制的弧形是由两条切线所决定。

​ 第 1 条切线：起始点和控制点1决定的直线。

​ 第 2 条切线：控制点1 和控制点2决定的直线。

​ 其实绘制的圆弧就是与这两条直线相切的圆弧。



(3)绘制文字 ---canvas绘图以左上角的（0， 0）为基准原点 但是文字以基线为基点

ctx.font = '100px 宋体‘’ // 设置字体大小 型号  
ctx.strokeText(‘abc’, x, y) // 实心  
ctx.fillText(‘abc’, x, y) // 空心  
ctx.stroke()

# 6.视频播放器

视频可以理解为一系列连续的图片，<video>标签的使用方法与<img>标签非常相似，

## **6.1具体语法如下所示**

(controls="controls"为视频提供播放控件↓)

<video src="视频文件路径“ controls="controls">

您的浏览器不支持video标签 //在浏览器不支持时显示

</video>

## 6.2<video>标签常用属性

|  |  |  |
| --- | --- | --- |
| **属性** | **允许取值** | **取值说明** |
| **autoplay** | autoplay | 如果出现该属性，则视频在就绪后马上播放。 |
| **controls** | controls | 如果出现该属性，则向用户显示控件，比如播放按钮。 |
| **height** | pixels | 设置视频播放器的高度。 |
| **loop** | loop | 如果出现该属性，则当媒体文件播放完后再次开始播放。 |
| **preload** | preload | 如果出现该属性，则视频在页面加载时进行加载，并预备播放。如果使用 "autoplay"，则忽略该属性。 |
| **src** | url | 要播放的视频的 URL。 |
| **width** | pixels | 设置视频播放器的宽度。 |

------<source>标签，用于指定多个备用的不同格式的文件的路径，语法如下所示：<source>标签对于音频文件同样适用，需要把video改成audio。

<video controls>

<source src="视频文件地址" type="媒体文件类型/格式">

<source src="视频文件地址" type="媒体文件类型/格式">

……

</video>

## 6.3Video对象的常用方法

|  |  |
| --- | --- |
| **方法** | **描述** |
| **load()** | 加载媒体文件，为播放做准备。通常用于播放前的预加载，也会用于重新加载媒体文件。 |
| **play()** | 播放媒体文件。如果视频没有加载，则加载并播放；如果视频是暂停的，则变为播放。 |
| **pause()** | 暂停播放媒体文件。 |
| **canPlayType()** | 测试浏览器是否支持指定的媒体类型。 |

## 6.4Video对象的常用方法和常用属性

|  |  |
| --- | --- |
| **属性** | **描述** |
| **currentSrc** | 返回当前视频的 URL。 |
| **currentTime** | 设置或返回视频中的当前播放位置（以秒计）。 |
| **duration** | 返回视频的长度（以秒计）。 |
| **ended** | 返回视频的播放是否已结束。 |
| **error** | 返回表示视频错误状态的 MediaError 对象。 |
| **paused** | 设置或返回视频是否暂停。 |
| **muted** | 设置或返回是否关闭声音。 |
| **volume** | 设置或返回视频的音量。 |
| **height** | 设置或返回视频的高度值。 |
| **width** | 设置或返回视频的宽度值。 |

## 6.5Video对象的常用事件

|  |  |
| --- | --- |
| **事件** | **描述** |
| **play** | 当执行方法play()时触发。 |
| **playing** | 正在播放时触发。 |
| **pause** | 当执行了方法pause()时触发。 |
| **timeupdate** | 当播放位置被改变时触发。 |
| **ended** | 当播放结束后停止播放时触发。 |
| **waiting** | 在等待加载下一帧时触发。 |

# 7.音频

HTML5中提供<audio>标签来定义Web上的声音文件或音频流.

## 7.1语法

<audio src="音频文件路径" controls>您的浏览器不支持audio标签</audio>

## 7.2< audio >标签的常用属性如下表所示。

|  |  |  |
| --- | --- | --- |
| **属性** | **允许取值** | **取值说明** |
| **autoplay** | autoplay | 如果出现该属性，则视频在就绪后马上播放。 |
| **controls** | controls | 如果出现该属性，则向用户显示控件，比如播放按钮。 |
| **loop** | loop | 如果出现该属性，则当媒体文件播放完后再次开始播放。 |
| **preload** | preload | 如果出现该属性，则视频在页面加载时进行加载，并预备播放。如果使用 "autoplay"，则忽略该属性。 |
| **src** | url | 要播放的视频的 URL。 |

------多个音频源使用<source>标签来定义，语法如下所示.

<audio controls>

<source src="音频文件路径" type="audio/格式">

<source src="音频文件路径" type="audio/格式">

您的浏览器不支持audio标签

</audio>

## 7.3Audio对象的常用方法

|  |  |
| --- | --- |
| **方法** | **描述** |
| **load()** | 加载媒体文件，为播放做准备。通常用于播放前的预加载，也会用于重新加载媒体文件。 |
| **play()** | 播放媒体文件。如果视频没有加载，则加载并播放；如果视频是暂停的，则变为播放。 |
| **pause()** | 暂停播放媒体文件。 |
| **canPlayType()** | 测试浏览器是否支持指定的媒体类型。 |

## 7.4Audio对象的常用属性

|  |  |
| --- | --- |
| **属性** | **描述** |
| **currentSrc** | 返回当前视频的 URL。 |
| **currentTime** | 设置或返回视频中的当前播放位置（以秒计）。 |
| **duration** | 返回视频的长度（以秒计）。 |
| **ended** | 返回视频的播放是否已结束。 |
| **error** | 返回表示视频错误状态的 MediaError 对象。 |
| **paused** | 设置或返回视频是否暂停。 |
| **muted** | 设置或返回是否关闭声音。 |
| **volume** | 设置或返回视频的音量。 |

## 7.5Audio对象的常用事件

|  |  |
| --- | --- |
| **事件** | **描述** |
| **play** | 当执行方法play()时触发。 |
| **playing** | 正在播放时触发。 |
| **pause** | 当执行了方法pause()时触发。 |
| **timeupdate** | 当播放位置被改变时触发。 |
| **ended** | 当播放结束后停止播放时触发。 |
| **waiting** | 在等待加载下一帧时触发。 |

# 8.图标库

<http://fontawesome.dashgame.com/#basic>

## 8.1基本图标

您可以将Font Awesome图标使用在几乎任何地方，只需要使用CSS前缀 fa ，再加上图标名称。 Font Awesome是为使用内联元素而设计的。我们通常更喜欢使用 <i> ，因为它更简洁。 但实际上使用 <span> 才能更加语义化。

<i class="fa fa-camera-retro"></i> fa-camera-retro

如果您修改了图标容器的字体大小，图标大小会随之改变。同样的变化也会发生在颜色、阴影等其它任何CSS支持的效果上。

## 8.2大图标

使用 fa-lg (33%递增)、fa-2x、 fa-3x、fa-4x，或者 fa-5x 类 来放大图标。

<i class="fa fa-camera-retro fa-lg"></i> fa-lg  
<i class="fa fa-camera-retro fa-2x"></i> fa-2x  
<i class="fa fa-camera-retro fa-3x"></i> fa-3x  
<i class="fa fa-camera-retro fa-4x"></i> fa-4x  
<i class="fa fa-camera-retro fa-5x"></i> fa-5x

如果图标的底部和顶部被截断了，您需要增加行高来解决此问题。

## 8.3固定宽度

使用 fa-fw 可以将图标设置为一个固定宽度。主要用于不同宽度图标无法对齐的情况。 尤其在列表或导航时起到重要作用。

<div class="list-group">  
  <a class="list-group-item" href="#"><i class="fa fa-home fa-fw"></i>&nbsp; Home</a>  
  <a class="list-group-item" href="#"><i class="fa fa-book fa-fw"></i>&nbsp; Library</a>  
  <a class="list-group-item" href="#"><i class="fa fa-pencil fa-fw"></i>&nbsp; Applications</a>  
  <a class="list-group-item" href="#"><i class="fa fa-cog fa-fw"></i>&nbsp; Settings</a>  
</div>

## 8.4用于列表

使用 fa-ul 和 fa-li 便可以简单的将无序列表的默认符号替换掉。

<ul class="fa-ul">  
  <li><i class="fa-li fa fa-check-square"></i>List icons</li>  
  <li><i class="fa-li fa fa-check-square"></i>can be used</li>  
  <li><i class="fa-li fa fa-spinner fa-spin"></i>as bullets</li>  
  <li><i class="fa-li fa fa-square"></i>in lists</li>  
</ul>

## 8.5边框与对齐

使用 fa-border 以及 pull-right 或 pull-left 可以轻易构造出引用的特殊效果。

<i class="fa fa-quote-left fa-3x pull-left fa-border"></i>  
...tomorrow we will run faster, stretch out our arms farther...  
And then one fine morning— So we beat on, boats against the  
current, borne back ceaselessly into the past.

## 8.6动画

使用 fa-spin 类来使任意图标旋转，现在您还可以使用 fa-pulse 来使其进行8方位旋转。尤其适合 fa-spinner、fa-refresh 和 fa-cog 。

<i class="fa fa-spinner fa-spin"></i>

<i class="fa fa-circle-o-notch fa-spin"></i> <i class="fa fa-refresh fa-spin"></i>

<i class="fa fa-cog fa-spin"></i>

<i class="fa fa-spinner fa-pulse"></i>

 CSS3动画不支持IE8-IE9。

## 8.7组合使用

如果想要将多个图标组合起来，使用 fa-stack 类作为父容器， fa-stack-1x 作为正常比例的图标， fa-stack-2x 作为大一些的图标。还可以使用 fa-inverse 类来切换图标颜色。

# 9.动画库

## 9.1 Animate.css动画库的简介

Animate.css 是一个来自国外的 CSS3 动画库，它预设了抖动（shake）、闪烁（flash）、弹跳（bounce）、翻转（flip）、旋转（rotateIn/rotateOut）、淡入淡出（fadeIn/fadeOut）等多达 60 多种动画效果，几乎包含了所有常见的动画效果。 浏览器兼容于支持 CSS3 animate 属性的浏览器，他们分别是：IE10+、Firefox、Chrome、Opera、Safari。

## 9.2 Animate.css动画库的使用

（1）首先引入animate css文件

查询地址： <https://daneden.github.io/animate.css/>

<head>

<link rel="stylesheet" href="animate.min.css"></head>

1. 给指定的元素加上指定的动画样式名

查询地址：<https://daneden.github.io/animate.css/> ----按F12查询动画类名

<div class="animated bounceOutLeft"></div>

这里包括两个class名，第一个是基本的，必须添加的样式名，任何想实现的元素都得添加这个。第二个是指定的动画样式名。

（3）如果说想给某个元素动态添加动画样式，可以通过jquery来实现：

$('#select).addClass('animated bounceOutLeft');

这里包括了jquery里的选择器 + addclass需要添加的动画类名

你也可以通过 JavaScript 或 jQuery 给元素添加这些 class，比如：

$(function(){

$('#jq22').addClass('animated bounce');

});

有些动画效果最后会让元素不可见，比如淡出、向左滑动等等，可能你又需要将 class 删除，比如：

$(function(){

$('#jq22').addClass('animated bounce');

setTimeout(function(){

$('#jq22').removeClass('bounce');

}, 1000);

});

animate.css 的默认设置也许有些时候并不是我们想要的，所以你可以重新设置，比如：

#jq22{

animate-duration: 2s; //动画持续时间

animate-delay: 1s; //动画延迟时间

animate-iteration-count: 2; //动画执行次数

# 10.响应式----媒体查询、栅格系统

<https://www.runoob.com/bootstrap/bootstrap-grid-system.html>

## 10.1媒体查询(@media 查询)

@media 查询，可以针对不同的媒体类型定义不同的样式。

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。

@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。

# 11.栅格系统

## 11.1什么是 Bootstrap 网格系统（Grid System）？

Bootstrap 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

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 | | | | | | | | | | | |

## 11.2网格选项

下表总结了 Bootstrap 网格系统如何跨多个设备工作：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **超小设备手机（<768px）** | **小型设备平板电脑（≥768px）** | **中型设备台式电脑（≥992px）** | **大型设备台式电脑（≥1200px）** |
| 网格行为 | 一直是水平的 | 以折叠开始，断点以上是水平的 | 以折叠开始，断点以上是水平的 | 以折叠开始，断点以上是水平的 |
| 最大容器宽度 | None (auto) | 750px | 970px | 1170px |
| Class 前缀 | **.col-xs-** | **.col-sm-** | **.col-md-** | **.col-lg-** |
| 列数量和 | 12 | 12 | 12 | 12 |
| 最大列宽 | Auto | 60px | 78px | 95px |
| 间隙宽度 | 30px （一个列的每边分别 15px） | 30px （一个列的每边分别 15px） | 30px （一个列的每边分别 15px） | 30px （一个列的每边分别 15px） |
| 可嵌套 | Yes | Yes | Yes | Yes |
| 偏移量 | Yes | Yes | Yes | Yes |
| 列排序 | Yes | Yes | Yes | Yes |

## 11.3基本的网格结构:container→row→col

下面是 Bootstrap 网格的基本结构：

<div class="container">

<div class="row">

<div class="col-\*-\*"></div>

<div class="col-\*-\*"></div>

</div>

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

## 11.4偏移列

使用 .col-md-offset-\* 类可以将列向右侧偏移。这些类实际是通过使用 \* 选择器为当前元素增加了左侧的边距（margin）。例如，.col-md-offset-4 类将 .col-md-4 元素向右侧偏移了4个列（column）的宽度。

## 11.5嵌套列

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）的个数不能超过12（其实，没有要求你必须占满12列）。

## 11.6列排序

通过使用 .col-md-push-\* 和 .col-md-pull-\* 类就可以很容易的改变列（column）的顺序。

# 12.Flex布局

参考资料: <http://www.ruanyifeng.com/blog/2015/07/flex-grammar.html>

## 12.1什么是Flex?

Flex 是 Flexible Box 的缩写，意为"弹性布局"，用来为盒状模型提供最大的灵活性。

任何一个容器都可以指定为 Flex 布局。

.box{

display: flex;

}

行内元素也可以使用 Flex 布局。

.box{

display: inline-flex;

}

Webkit 内核的浏览器，必须加上-webkit前缀。

.box{

display: -webkit-flex; /\* Safari \*/

display: flex;

}

## 12.2基本概念

容器默认存在两根轴：水平的主轴（main axis）和垂直的交叉轴（cross axis）。

## 12.3容器的属性

flex-direction

flex-wrap

flex-flow

justify-content

align-items

align-content

flex-direction属性

flex-direction属性决定主轴的方向（即项目的排列方向）。

.box {

flex-direction: row | row-reverse | column | column-reverse;

}

它可能有四个值:

row（默认值）：主轴为水平方向，起点在左端。

row-reverse：主轴为水平方向，起点在右端。

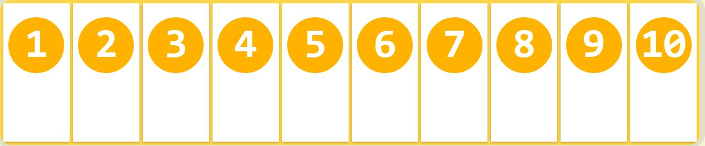
column：主轴为垂直方向，起点在上沿。

column-reverse：主轴为垂直方向，起点在下沿。

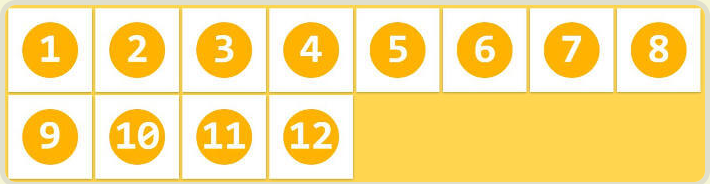
flex-wrap属性

设置换行,它可能有三个值:

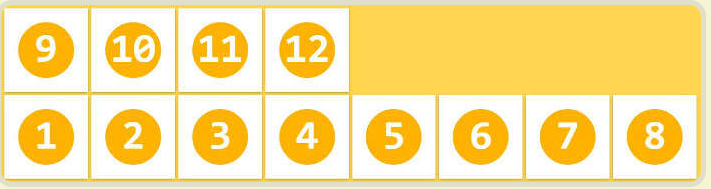
(1) nowrap（默认）：不换行。



(2) wrap：换行，第一行在上方。



(3) wrap-reverse：换行，第一行在下方。



3.3 flex-flow

flex-flow属性是flex-direction属性和flex-wrap属性的简写形式，默认值为row nowrap

.box {

flex-flow: <flex-direction> || <flex-wrap>;

}

3.4 justify-content属性

justify-content属性定义了项目在主轴上的对齐方式。

.box {

justify-content: flex-start | flex-end | center | space-between | space-around;

}

flex-start（默认值）：左对齐

flex-end：右对齐

center： 居中

space-between：两端对齐，项目之间的间隔都相等。

space-around：每个项目两侧的间隔相等。所以，项目之间的间隔比项目与边框的间隔大一倍。

3.5 align-items属性

它可能取5个值。具体的对齐方式与交叉轴的方向有关，下面假设交叉轴从上到下。

flex-start：交叉轴的起点对齐。

flex-end：交叉轴的终点对齐。

center：交叉轴的中点对齐。

baseline: 项目的第一行文字的基线对齐。

stretch（默认值）：如果项目未设置高度或设为auto，将占满整个容器的高度。

3.6 align-content属性

align-content属性定义了多根轴线的对齐方式。如果项目只有一根轴线，该属性不起作用。

该属性可能取6个值。

flex-start：与交叉轴的起点对齐。

flex-end：与交叉轴的终点对齐。

center：与交叉轴的中点对齐。

space-between：与交叉轴两端对齐，轴线之间的间隔平均分布。

space-around：每根轴线两侧的间隔都相等。所以，轴线之间的间隔比轴线与边框的间隔大一倍。

stretch（默认值）：轴线占满整个交叉轴。

## 12.4项目的属性

设置项目的属性有6个

order

flex-grow

flex-shrink

flex-basis

flex

align-self

4.1 order属性

属性定义项目的排列顺序。数值越小，排列越靠前，默认为0。

4.2 flex-grow属性

属性定义项目的放大比例，默认为0，即如果存在剩余空间，也不放大。

4.3 flex-shrink属性

属性定义了项目的缩小比例，默认为1，即如果空间不足，该项目将缩小。

4.4 flex-basis属性

属性定义了在分配多余空间之前，项目占据的主轴空间（main size）。浏览器根据这个属性，计算主轴是否有多余空间。它的默认值为auto，即项目的本来大小。

4.5 flex属性

flex属性是flex-grow, flex-shrink和flex-basis的简写，默认值为0 1 auto。后两个属性可选。

4.6 align-self属性

属性允许单个项目有与其他项目不一样的对齐方式，可覆盖align-items属性。默认值为auto，表示继承父元素的align-items属性，如果没有父元素，则等同于stretch

# 13.bootstrap

参考网站:

<https://v3.bootcss.com/css/#top>

<https://www.runoob.com/bootstrap/bootstrap-grid-system.html>

[https://bootswatch.com](https://bootswatch.com/)

## 13.1安装和依赖

安装网站: <https://v3.bootcss.com/>

首先安装bootstrap包,并且引入bootstrap的css的文件和js文件

依赖:

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

<script src="jquery-2.js" type="text/javascript" charset="utf-8"></script>

<script src="bootstrap/js/bootstrap.js" type="text/javascript" charset="utf-8"></script>

因为bootstrap是基于Jquery的框架,所以一定要在引入bootstrap.js 之前引入Jquery

//<!-- 兼容IE9 -->

<!-- <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> -->

## 13.2使用

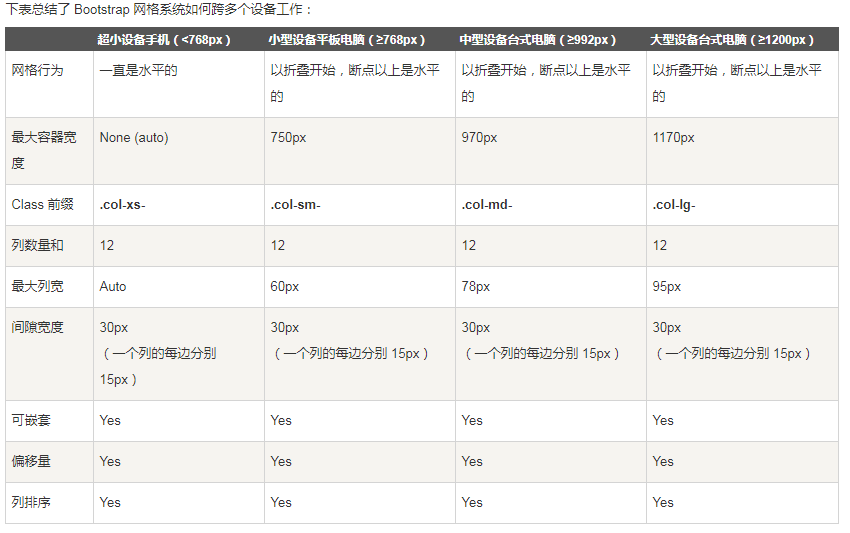
使用bootstrap的时候,只需要引用它内置的类名或标签名即可使用

例如:

.container类用于固定宽度并支持响应式布局的容器。

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

## 13.3网格系统



基本的网格结构

<div class="container">

<div class="row">

<div class="col-\*-\*"></div>

<div class="col-\*-\*"></div>

</div>

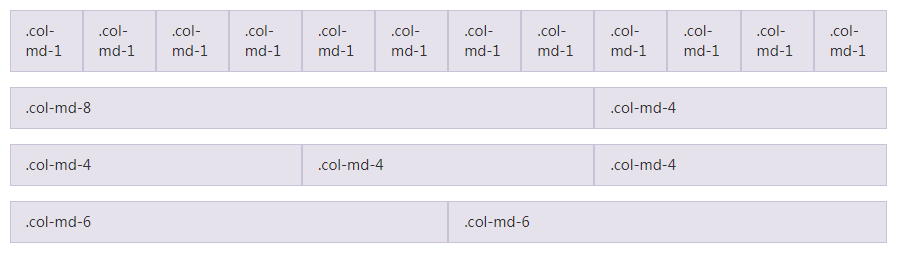
<div class="row">...</div>

</div>

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

所有列（column）必须放在 .row内。

实例:



<div class="row">

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

<div class="col-md-1">.col-md-1</div>

</div>

<div class="row">

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

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

</div>

<div class="row">

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

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

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

</div>

<div class="row">

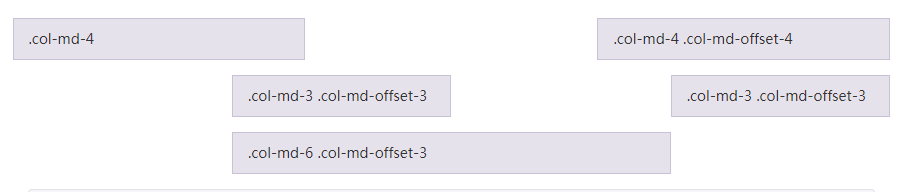
<div class="col-md-6">.col-md-6</div>

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

</div>

## 13.3.1 列偏移

使用.col-md-offset-\*类可以将列向右侧偏移。这些类实际是通过使用选择器为当前元素增加了左侧的边距（margin）。例如，.col-md-offset-4类将.col-md-4元素向右侧偏移了4个列（column）的宽度。



## 13.3.2 嵌套列

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）的个数不能超过12（其实，没有要求你必须占满12列）。



## 13.3.3列排序

通过使用.col-md-push-\*和.col-md-pull-\*类就可以很容易的改变列（column）的顺序。



## 13.4组件库

无数可复用的组件，包括字体图标、下拉菜单、导航、警告框、弹出框等更多功能。

## 13.4.1使用方法

出于性能的考虑，所有图标都需要一个基类和对应每个图标的类。把下面的代码放在任何地方都可以正常使用。注意，为了设置正确的内补（padding），务必在图标和文本之间添加一个空格。

实例:

<button type="button" class="btn btn-default" aria-label="Left Align">

<span class="glyphicon glyphicon-align-left" aria-hidden="true"></span>

</button>

<button type="button" class="btn btn-default btn-lg">

<span class="glyphicon glyphicon-star" aria-hidden="true"></span> Star

</button>

导航栏,下拉菜单亦是如此,只需要引用相对应的类名或标签吗即可使用

## 13.5主题切换

下载网站: [https://bootswatch.com](https://bootswatch.com/)

主题的使用只需要下载然后替换原来的CSS文件即可.替换好之后使用相应类名即可展现该主题的效果