**BOOTSTRAP第一天笔记**

**响应式页面**

**响应式页面概述**

**1. 什么是响应式页面**

响应式网页（RWD，Responsive Web Design），2010年5月由Ethan Marcotte提出。简而言之，就是一个网站能够针对多个终端提供不同的显示样式，而不是针对每个终端制作一个特定的版本。

由于智能手机和平板电脑的兴起，网站页面的制作不仅要满足PC端浏览器的显示，还要满足智能手机浏览器的显示或者平板电脑浏览器的显示。（**PC端、智能手机或平板电脑的浏览器之间的区别，主要集中在分辨率的不同。**）

**2. 为什么需要响应式页面**

想要同时满足PC端、智能手机和平板电脑浏览器的正常显示，实际上有以下两种实现方式：

a. 分别针对PC端、智能手机和平板电脑的浏览器制作特定的页面。例如 <https://www.tmall.com/>

* 优点：用户体验好，适合复杂页面。
* 缺点：工作量大，不够灵活。

b. 制作一个页面同时满足PC端、智能手机和平板电脑的浏览器显示。例如 <http://www.intel.cn/>

* 优点：比较灵活，一个页面适应不同终端。
* 缺点：页面加载时间长，效率降低。

如果PC端显示的页面与智能手机或平板电脑显示的页面不同时，建议选择 *a* 方案。比如智能手机端的可以利用 GPS 定位当前位置等不同于PC端的功能。但大多数网站的页面，无论是PC端还是智能手机或平板电脑显示的内容基本相同，建议选择 *b* 方案。

**3. 响应式页面的构成**

目前实现响应式页面可以通过以下两种方式：

a. 使用框架实现，例如 [Bootstrap](https://getbootstrap.com) 等。这种方式开发响应式页面更快速、简单，但由于CSS样式是集成式的，导致页面样式相似，没有特点。

b. 手写方式实现。这种方式开发响应式页面速度比较慢、代码多，但可以根据网站特点定制CSS样式。

手写方式需要使用到以下几种技术：

* 流式网格布局
* CSS3 的媒体查询
* 相对单位替换绝对单位

**测试响应式页面**

**1. 如何测试响应式页面**

测试响应式页面可以通过以下三种方式：

* 通过真实设备进行测试。这种方式进行测试效果最好，但也是最复杂，并且需要不同分辨率的设备，成本较高。
* 通过第三方模拟器进行测试。这种方式一般都是针对智能手机终端研发工具自带，需要在电脑安装不同开发工具，比较麻烦，并且性能较差。或者通过在线测试网站进行测试，测试周期较长。
* 通过浏览器自带的设备模拟器进行测试。这种方式是最方便的，主流浏览器都自带了设备模拟器，并且测试效果较好。

**2. 使用浏览器测试**

以 Chrome 浏览器为例。

打开 Chrome 浏览器，点击右上角的菜单，选择“更多工具”中的“开发者工具”。（**Windows操作系统版本的 Chrome 浏览器可以通过 F12 快捷键直接打开。**）

**值得注意的是：**

* 每次更换测试设备时，需要重新刷新页面显示。
* 有些功能无法正常测试，例如电话薄、摄像头等。

**3. 使用第三方工具测试**

**4. 使用真实设备测试**

**编写响应式页面**

**1. viewport元元素**

**1）什么是 viewport ？**

viewport 被翻译为“视区”，表示页面的可视区域。通过 HTML 页面的 meta 元元素进行设置，是响应式页面设计的必备内容。

移动端浏览器在一个通常比屏幕更宽的虚拟“窗口”（视口）中渲染页面，从而无序将所有页面都压缩进小屏幕里（那样会把很多没有针对移动端进行优化的站点打乱）。用户可以通过平移和缩放来浏览页面的不同区域。

通过设置 viewport 元元素可以允许开发者控制显示的尺寸及比例。如今大部分移动端浏览器都支持 viewport 元元素，尽管它**不是 web 标准的一部分**。

**2）设置 viewport**

一个典型的针对移动端优化的站点包含类似下面的内容：

<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">

viewport 元元素的属性如下表：

| **属性名称** | **说明** | **可选值** |
| --- | --- | --- |
| width | 设置窗口的宽度 | 像素值或device-width（表示100%时屏幕宽度的像素值） |
| height | 设置窗口的高度 | 一般不指定 |
| initial-scale | 设置初始时的缩放倍率 |  |
| minimum-scale | 设置允许的最小缩放倍率 |  |
| maximum-scale | 设置允许的最大缩放倍率 |  |
| user-scalable | 是否允许用户手动缩放 | 1或yes表示允许；0或no表示不允许 |

meta 元元素建议设定在 head 元素中的最前面。

**2. 使用相对单位**

**1）使用相对宽度值**

通过学习 CSS 知道设置宽度可以使用 px 像素值和百分值。使用像素值设置宽度是固定值，不能随着屏幕大小变化而变化。实现响应式页面的宽度只能使用百分值或者auto。

**如何从固定布局修改为百分比布局**

在《无懈可击的 Web 设计》一书中，提供了一个简易可行的公式：

目标元素宽度 / 上下文元素宽度 = 百分比宽度

以下代码是使用像素值设置 HTML 页面元素：

div {

width : 500px;

height : auto;

border : 1px solid black;

}

p {

width : 480px;

padding-left: 10px;

}

将上述像素值修改为百分比：

div {

width : 60%;

height : auto;

border : 1px solid black;

}

p {

width : 96%; /\* 480/500=96% \*/

padding-left: 2%; /\* 10/500=2% \*/

}

**2）用 em 替换 px**

之前使用 em 替代 px 设置字体，主要是为了解决老版本的 IE 浏览器无法缩放以像素值为单位的文字。

目前使用 em 替代 px 设置字体，主要的目的具有以下两个：

* 实现老版本 IE 浏览器可以缩放文字。
* 更好地实现响应式页面，使页面效果更好。

em 的实际大小事相对于上下文的字体大小而言的，例如 1em 表示与上下文字体相同大小。em 单位设置字体同样可以通过以下公式进行计算：

目标元素尺寸 / 上下文元素尺寸 = 百分比尺寸

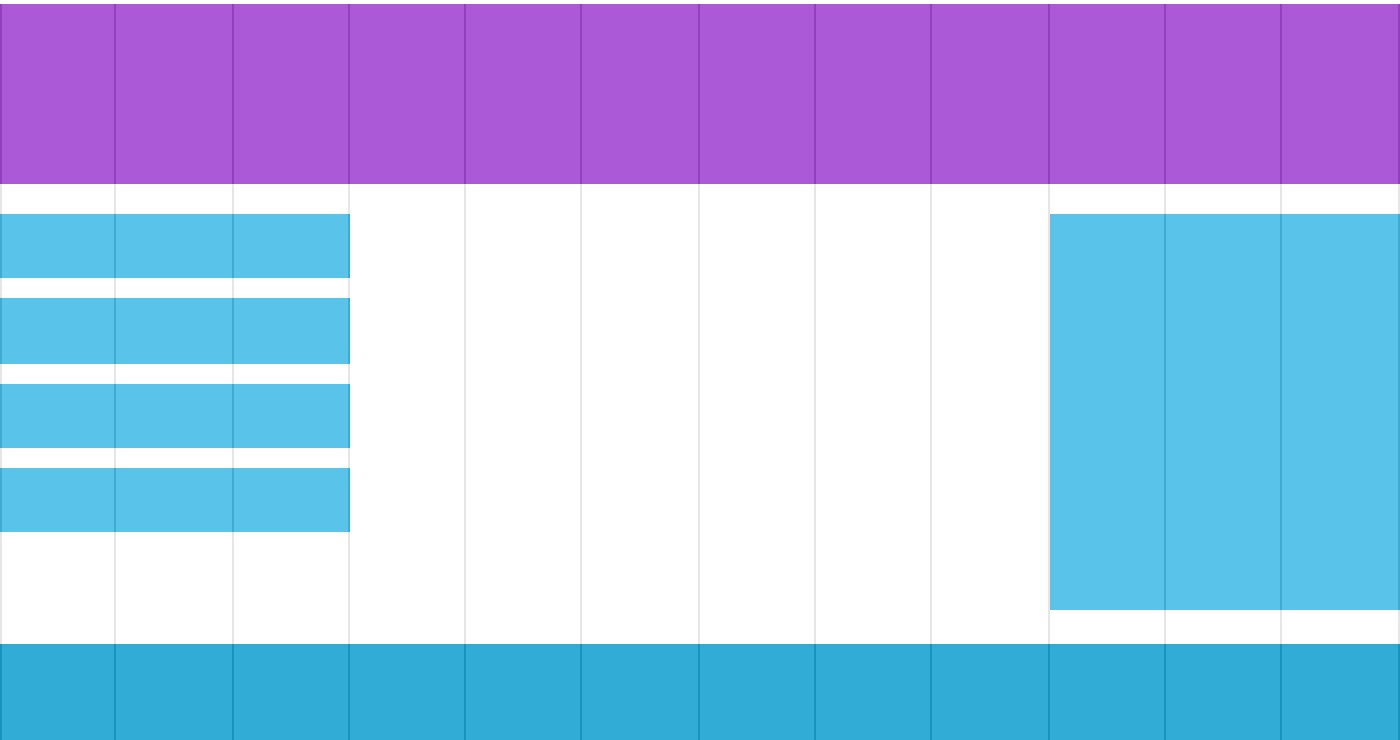
**值得注意的是：**浏览器显示默认文字大小都是 16 像素。

**3. 流式布局**

**1）什么是流式布局**

所谓流式布局，就是页面元素的宽度按照屏幕进行适配调整。简单来说，就是 HTML 页面中的元素会根据分辨率的不同而变化大小，但位置并不会有任何变化。

这种布局的主要问题就是，如果屏幕尺度跨度太大，那么在相对其原始设计而言过小或过大的屏幕上不能正常显示。



**2）创建流式布局**

流式布局通常被分为 12 列。将 HTML 页面的宽度设置为 100%，这样 HTML 页面会根据浏览器窗口大小的变化而自动伸缩。

* 计算每列所占的百分比：100% / 12 列 = 8.33%

根据以上的计算结果，设置 CSS 的 class：

.col-1 {width: 8.33%;}

.col-2 {width: 16.66%;}

.col-3 {width: 25%;}

.col-4 {width: 33.33%;}

.col-5 {width: 41.66%;}

.col-6 {width: 50%;}

.col-7 {width: 58.33%;}

.col-8 {width: 66.66%;}

.col-9 {width: 75%;}

.col-10 {width: 83.33%;}

.col-11 {width: 91.66%;}

.col-12 {width: 100%;}

* 将所有列设置为浮动（一般为左浮动）

[class\*="col-"] {

float: left;

}

* 将 HTML 页面中所有的元素 box-sizing 设置为 border-box。
  + 当 box-sizing 的值为 content-box（默认值），标准盒模型。

尺寸计算公式：width = 内容的宽度，height = 内容的高度。

宽度和高度都不包含内容的边框（border）和内边距（padding）。

div {

width : 300px;

height : 200px;

border : 10px solid black;

background : deepskyblue;

}

* + 当 box-sizing 的值为 border-box，IE 怪异模式（Quirks mode）使用的 盒模型 。

尺寸计算公式：width = border + padding + 内容的宽度，height = border + padding + 内容的高度。

**注意：**外边距和边框将会包括在盒子中。

div {

box-sizing: border-box;

width : 300px;

height : 200px;

border : 10px solid black;

background : deepskyblue;

}

根据上述流式布局的内容，实现以下案例：

<!DOCTYPE html>

<html>

<head>

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

<style>

\* {

box-sizing: border-box;

}

[class\*="col-"] {

float: left;

padding: 15px;

}

.col-1 {

width: 8.33%;

}

.col-2 {

width: 16.66%;

}

.col-3 {

width: 25%;

}

.col-4 {

width: 33.33%;

}

.col-5 {

width: 41.66%;

}

.col-6 {

width: 50%;

}

.col-7 {

width: 58.33%;

}

.col-8 {

width: 66.66%;

}

.col-9 {

width: 75%;

}

.col-10 {

width: 83.33%;

}

.col-11 {

width: 91.66%;

}

.col-12 {

width: 100%;

}

html {

font-family: "Lucida Sans", sans-serif;

}

header {

background-color: #9933cc;

color: #ffffff;

padding: 15px;

}

aside ul {

list-style-type: none;

margin: 0;

padding: 0;

}

aside li {

padding: 8px;

margin-bottom: 7px;

background-color: #33b5e5;

color: #ffffff;

box-shadow: 0 1px 3px rgba(0, 0, 0, 0.12), 0 1px 2px rgba(0, 0, 0, 0.24);

}

aside li:hover {

background-color: #0099cc;

}

</style>

</head>

<body>

<header>

<h1>Chania</h1>

</header>

<aside class="col-3">

<ul>

<li>The Flight</li>

<li>The City</li>

<li>The Island</li>

<li>The Food</li>

</ul>

</aside>

<article class="col-9">

<h2>The City</h2>

<p>Chania is the capital of the Chania region on the island of Crete. The city can be divided in two parts, the old

town and the modern city.</p>

<p>Resize the browser window to see how the content respond to the resizing.</p>

</article>

</body>

</html>

**4. 响应式图片**

实现图片的显示根据浏览器窗口大小的变化而变化，相对比较简单。只需要将图片的 width 设置为 100% 即可。

img {

width: 100%;

height: auto;

}

**但这样设置后，图片的显示可能会比实际大小更大。解决这个问题可以利用 max-width 替代 width 属性即可。**

img {

max-width: 100%;

height: auto;

}

这样设置后，图片的显示永远不会比实际大小更大了。

**5. CSS媒体查询**

想要满足更复杂的情况，需要使用 CSS3 中提供的媒体查询进行解决。

媒体查询 包含了一个媒体类型和至少一个使用如宽度、高度和颜色等媒体属性来限制样式表范围的表达式。CSS3加入的媒体查询使得无需修改内容便可以使样式应用于某些特定的设备范围。

**媒体查询语法**

媒体查询 包含 一个媒体类型、逻辑操作符和一个或多个媒体属性。如果媒体查询中的媒体类型与文档要展示的设备相符则查询结果为真，并且媒体查询中的所有表达式为真。

媒体查询具体的写法有以下两种方式：

* link 元素中的 CSS 媒体查询。

<link rel="stylesheet" media="mediatype and|not|only (media feature)" href="mystylesheet.css">

这种方式需要针对不同设备编写不同的 CSS 样式表。

* 样式表中的 CSS 媒体查询。

@media mediatype and|not|only (media feature) {

CSS-Code;

}

这种方式只需要编写一个 CSS 样式表，在样式表中根据不同的设备设置不同的样式。（**g更常用**）

**媒体类型**

| **值** | **描述** |
| --- | --- |
| all | 用于所有设备 |
| print | 用于打印机和打印预览 |
| screen | 用于电脑屏幕，平板电脑，智能手机等 |
| speech | 用于屏幕阅读器等发声设备 |

媒体类型更常用的值为 screen。

**逻辑操作符**

| **值** | **描述** |
| --- | --- |
| and | 用于把多个 媒体属性 组合起来，合并到同一条媒体查询中。只有当每个属性都为真时，这条查询的结果才为真. |
| not | 用于对一条媒体查询的结果进行取反。 |
| only | 表示仅在媒体查询匹配成功的情况下应用指定样式。可以通过它让选中的样式在老式浏览器中不被应用。 |

也可以将多个媒体查询以逗号分隔放在一起；只要其中任何一个为真，整个媒体语句就返回真。相当于 or 操作符。

| **值** | **描述** |
| --- | --- |
| width | 定义输出设备中的页面可见区域宽度 |
| max-width | 定义输出设备中的页面最大可见区域宽度 |
| max-device-width | 定义输出设备的屏幕最大可见宽度 |
| min-width | 定义输出设备中的页面最小可见区域宽度 |
| min-device-width | 定义输出设备的屏幕最小可见宽度 |

**媒体查询示例**

**link 元素中的 CSS 媒体查询**

* HTML 代码

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>LINK元素方式的媒体查询</title>

<meta name="viewport" content="width=device-width,initial-scale=1"/>

<link rel="stylesheet" href="css/main.css">

<link rel="stylesheet" media="screen and (min-width:768px)" href="css/pc.css">

<link rel="stylesheet" media="screen and (max-width:768px)" href="css/phone.css">

</head>

<body>

<header>

<h1>Chania</h1>

</header>

<aside class="col-3">

<ul>

<li>The Flight</li>

<li>The City</li>

<li>The Island</li>

<li>The Food</li>

</ul>

</aside>

<article class="col-9">

<h2>The City</h2>

<p>Chania is the capital of the Chania region on the island of Crete. The city can be divided in two parts, the old

town and the modern city.</p>

<p>Resize the browser window to see how the content respond to the resizing.</p>

</article>

</body>

</html>

* 针对分辨率的宽度在 768px 之上的样式表

.col-1 {

width: 8.33%;

}

.col-2 {

width: 16.66%;

}

.col-3 {

width: 25%;

}

.col-4 {

width: 33.33%;

}

.col-5 {

width: 41.66%;

}

.col-6 {

width: 50%;

}

.col-7 {

width: 58.33%;

}

.col-8 {

width: 66.66%;

}

.col-9 {

width: 75%;

}

.col-10 {

width: 83.33%;

}

.col-11 {

width: 91.66%;

}

.col-12 {

width: 100%;

}

[class\*="col-"] {

float: left;

padding: 15px;

}

* 针对分辨率的宽度在 768px 之下的样式表

[class\*="col-"] {

width: 100%;

padding: 15px;

}

**样式表中的 CSS 媒体查询**

* HTML 代码

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>CSS样式表中的媒体查询</title>

<meta name="viewport" content="width=device-width,initial-scale=1"/>

<link rel="stylesheet" href="css/main.css">

<link rel="stylesheet" href="css/media.css">

</head>

<body>

<header>

<h1>Chania</h1>

</header>

<aside class="col-3">

<ul>

<li>The Flight</li>

<li>The City</li>

<li>The Island</li>

<li>The Food</li>

</ul>

</aside>

<article class="col-9">

<h2>The City</h2>

<p>Chania is the capital of the Chania region on the island of Crete. The city can be divided in two parts, the old

town and the modern city.</p>

<p>Resize the browser window to see how the content respond to the resizing.</p>

</article>

</body>

</html>

* CSS 代码

/\* For desktop: \*/

.col-1 {

width: 8.33%;

}

.col-2 {

width: 16.66%;

}

.col-3 {

width: 25%;

}

.col-4 {

width: 33.33%;

}

.col-5 {

width: 41.66%;

}

.col-6 {

width: 50%;

}

.col-7 {

width: 58.33%;

}

.col-8 {

width: 66.66%;

}

.col-9 {

width: 75%;

}

.col-10 {

width: 83.33%;

}

.col-11 {

width: 91.66%;

}

.col-12 {

width: 100%;

}

[class\*="col-"] {

float: left;

padding: 15px;

}

@media only screen and (max-width: 768px) {

/\* For mobile phones: \*/

[class\*="col-"] {

width: 100%;

}

}

**BOOTSTRAP**

**起步**

**1. 什么是BOOTSTRAP**

**1）BOOTSTRAP概述**

Bootstrap 是 Twitter 公司开发的一个基于 HTML、CSS、JavaScript 的技术框架，符合 HTML、CSS 规范，且代码简介、视觉优美。

Bootstrap 集合 HTML、CSS 和 JavaScript，使用了最新的浏览器技术，为实现快速开发提供了一套前端工具包。使用 Bootstrap 不仅可以构建出非常优雅的 HTML 页面，而且占用资源非常少，使用 gzip 压缩后大小仅用 10KB。并且 Bootstrap 在跨浏览器兼容方面表示也很好。

**2）选择BOOTSTRAP的理由**

* Bootstrap 的 HTML 是基于 HTML5 的最新技术。
* Bootstrap 可以快速实现响应式页面。
* Bootstrap 集成了非常友好的 CSS 样式表，对于非设计人员也可以制作出很漂亮的网页。

**3）BOOTSTRAP提供的功能**

根据 Bootstrap 官网提供的文档，可以知道 Bootstrap 分为以下几个模块：

* 全局 CSS 样式：提供了格栅系统、表格、表单、按钮等集成样式。
* 组件：提供了下拉菜单、输入框、导航、列表组等组件。
* 插件：提供了模态框、滚动监听、警告框、弹出框等插件。

**4）BOOTSTRAP的版本变化**

* 2011年8月，Twitter 推出了 Bootstrap 1 版本，该工具由 Twitter 的设计师 Mark Otto 和 Jacob Thornton 合作完成。
* 2012年1月，Twitter 推出了 Bootstrap 2 版本。相比 Bootstrap 1 版本，Bootstrap 2 添加了响应式设计、采用 12 栏栅格布局，并且清晰地划分出 CSS 布局、组件和插件等功能。
* 2013年8月，Twitter 推出了 Bootstrap 3 版本。在这个版本中，采用移动优先和更好的盒子模型等。并且官方不再支持 IE 7及以下版本的浏览器。
* 目前，Bootstrap 4 只是开发者预览版。

**2. 下载BOOTSTRAP**

**1）Bootstrap 官网及下载地址**

* Bootstrap 官网：<http://getbootstrap.com>
* Bootstrap 下载地址：[http://getbootstrap.com/getting-started/#download](http://getbootstrap.com/getting-started/" \l "download)

**2）Bootstrap 使用的两种方式**

* 将 Bootstrap 提供的压缩包下载，导入到工程目录中使用。
* 使用 CDN 加速服务，例如以下内容：

<!-- 新 Bootstrap 核心 CSS 文件 -->

<link rel="stylesheet" href="//cdn.bootcss.com/bootstrap/3.3.6/css/bootstrap.min.css">

<!-- 可选的Bootstrap主题文件（一般不用引入） -->

<link rel="stylesheet" href="//cdn.bootcss.com/bootstrap/3.3.6/css/bootstrap-theme.min.css">

<!-- jQuery文件。务必在bootstrap.min.js 之前引入 -->

<script src="//cdn.bootcss.com/jquery/1.11.3/jquery.min.js"></script>

<!-- 最新的 Bootstrap 核心 JavaScript 文件 -->

<script src="//cdn.bootcss.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>

**3）Bootstrap 目录结构及说明**

下载压缩包之后，将其解压缩到任意目录即可看到以下（压缩版的）目录结构：

bootstrap/

├── css/ 样式库

│ ├── bootstrap.css 样式文件

│ ├── bootstrap.css.map

│ ├── bootstrap.min.css 压缩后的样式文件

│ ├── bootstrap-theme.css 主题文件

│ ├── bootstrap-theme.css.map

│ └── bootstrap-theme.min.css 压缩后的主题文件

├── js/ 核心 js 文件

│ ├── bootstrap.js 核心 js 文件

│ └── bootstrap.min.js 压缩后的核心 js 文件

└── fonts/ 字体库

├── glyphicons-halflings-regular.eot

├── glyphicons-halflings-regular.svg

├── glyphicons-halflings-regular.ttf

├── glyphicons-halflings-regular.woff

└── glyphicons-halflings-regular.woff2

**3. 对浏览器的支持情况**

**1）目前支持的浏览器**

|  | **Chrome** | **Firefox** | **Internet Explorer** | **Opera** | **Safari** |
| --- | --- | --- | --- | --- | --- |
| Android | 支持 | 支持 | N/A | 不支持 | N/A |
| IOS | 支持 | N/A | N/A | 不支持 | 支持 |
| Mac OS X | 支持 | 支持 | N/A | 支持 | 支持 |
| Windows | 支持 | 支持 | 支持 | 支持 | 不支持 |

**2）IE 8及之前版本的兼容**

* 对 HTML 5 提供的新元素，需要通过 html5shiv.js 库进行兼容。
* 对 CSS 3 提供的媒体查询，需要通过 respond.js 库进行支持。

需要在使用 Bootstrap 框架的页面 head 元素中，插入以下内容：

<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->

<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->

<!--[if lt IE 9]>

<script src="//cdn.bootcss.com/html5shiv/3.7.2/html5shiv.min.js"></script>

<script src="//cdn.bootcss.com/respond.js/1.4.2/respond.min.js"></script>

<![endif]-->

**3）IE 兼容模式**

Bootstrap 不支持 IE 古老的兼容模式。为了让 IE 浏览器运行最新的渲染模式下，建议将此 <meta> 元素加入到页面中：

<meta http-equiv="X-UA-Compatible" content="IE=edge">

**4）国产浏览器高速模式**

国内浏览器厂商一般都支持兼容模式（即 IE 内核）和高速模式（即 webkit 内核），不幸的是，所有国产浏览器都是默认使用兼容模式，这就造成由于低版本 IE （IE8 及以下）内核让基于 Bootstrap 构建的网站展现效果很糟糕的情况。

将下面的 <meta> 元素加入到页面中，可以让部分国产浏览器默认采用高速模式渲染页面：

<meta name="renderer" content="webkit">

**4. box-sizing**

Bootstrap 默认使用的盒子模型是 border-box，可能会和一些第三方组件冲突。

为了避免 Bootstrap 设置的全局盒模型所带来的影响，可以重置单个页面元素或覆盖整个区域的盒模型。

* 覆盖单个页面元素

/\* 通过 CSS 代码覆盖单个页面元素的盒模型 \*/

.element {

-webkit-box-sizing: content-box;

-moz-box-sizing: content-box;

box-sizing: content-box;

}

* 重置整个区域

/\* 通过 CSS 代码重置整个区域 \*/

.reset-box-sizing,

.reset-box-sizing \*,

.reset-box-sizing \*:before,

.reset-box-sizing \*:after {

-webkit-box-sizing: content-box;

-moz-box-sizing: content-box;

box-sizing: content-box;

}

**5. 基本模板**

Bootstrap 提供了一个最基本的 HTML 模板，基于这个模板开始实现 Bootstrap 响应式页面。

<!DOCTYPE html>

<html lang="zh-CN">

<head>

<!-- 设置当前 HTML 页面的编码格式为 UTF-8 -->

<meta charset="utf-8">

<!-- 设置 IE 的兼容模式 -->

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<!-- 设置移动优先 -->

<meta name="viewport" content="width=device-width, initial-scale=1">

<!-- 上述3个meta标签必须放在最前面，任何其他内容都必须跟随其后！ -->

<title>Bootstrap最基本的HTML模板</title>

<!-- 导入 Bootstrap 框架的 CSS 文件 -->

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

<!--

html5shiv.js 文件解决 IE8及之前版本的浏览器支持 HTML 5 元素的问题。

respond.js 文件解决 IE8及之前版本的浏览器支持 CSS 3 的媒体查询问题。

-->

<!--[if lt IE 9]>

<script src="bootstrap/js/html5shiv.min.js"></script>

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

<![endif]-->

</head>

<body>

<h1>你好，世界！</h1>

<!--

由于 Bootstrap 是基于 jQuery 实现的核心 js 功能,

所以想要使用 Bootstrap 提供的插件功能,需要先导入 jQuery 文件。

-->

<script src="bootstrap/js/jquery-1.11.3.js"></script>

<!-- 导入 Bootstrap 框架的 js 文件 -->

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

</body>

</html>

**全局CSS样式**

**1. 概述**

**1）HTML 5 文档类型**

Bootstrap 使用到的某些 HTML 元素和 CSS 属性需要将页面设置为 HTML5 文档类型。

<!DOCTYPE html>

<html lang="zh-CN">

...

</html>

**2）移动设备优先**

Bootstrap 是移动设备优先的。为了确保适当的绘制和触屏缩放，需要在 <head> 之中添加 viewport 元元素。

<meta name="viewport" content="width=device-width, initial-scale=1">

在移动设备浏览器上，通过为 viewport设置 meta 属性为 user-scalable=no 可以禁用其缩放功能。这样禁用缩放功能后，用户只能滚动屏幕，就能让你的网站看上去更像原生应用的感觉。

<meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no">

**3）排版与链接**

Bootstrap 排版、链接样式设置了基本的全局样式。

* 为 body 元素设置 background-color: #fff;
* 使用 @font-family-base、@font-size-base 和 @line-height-base 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

**2. Normalize.css**

**1）CSS 重置样式库**

为了增强跨浏览器表现的一致性，Bootstrap 使用 Normalize.css（CSS 重置样式库）。

**2）什么是 Normalize.css**

Normalize.css 只是一个很小的 CSS 文件，但它在默认的 HTML 元素样式上提供了跨浏览器的高度一致性。相比于传统的 CSS reset，Normalize.css 是一种现代的、为 HTML5 准备的优质替代方案。

Normalize.css 的主要特点如下：

* 保护有用的浏览器默认样式而不是完全去掉它
* 一般化的样式：为大部分HTML元素提供
* 修复浏览器自身的bug并保证各浏览器的一致性
* 优化CSS可用性：用一些小技巧
* 解释代码：用注释和详细的文档来

**3）Normalize.css 与传统 Reset 的区别**

* Normalize.css 保护了有价值的默认值
* Normalize.css 修复了浏览器的bug
* Normalize.css 不会让你的调试工具变的杂乱
* Normalize.css 是模块化的
* Normalize.css 拥有详细的文档

**3. 布局容器**

Bootstrap 提供了以下两种布局容器：

* .container 类用于固定宽度并支持响应式布局的容器。

| **媒体查询** | **宽度值** |
| --- | --- |
| >1200px | 1170px |
| >992px | 970px |
| >768px | 750px |
| 小分辨率 | 100% |

* .container-fluid 类用于 100% 宽度，占据 viewport 的容器。

**值得注意的是：**这两种 容器类不能互相嵌套。

**4. 按钮**

**1）作为按钮的元素**

Bootstrap 利用 <a>、<button>和<input>元素作为按钮样式。

<div class="container">

<div class="row">

<a class="btn btn-default" href="#" role="button">Link</a>

<button class="btn btn-default" type="submit">Button</button>

<input class="btn btn-default" type="button" value="Input">

<input class="btn btn-default" type="submit" value="Submit">

</div>

</div>

**值得注意的是：**

1. 导航和导航条组件只支持 <button> 元素。
2. 如果 <a> 元素被作为按钮使用，务必为其设置 role="button" 属性。
3. 建议尽可能使用 <button> 元素。

**2）预定义样式**

Bootstrap 为按钮预定义了很多样式，具体样式说明如下：

| **样式名称** | **描述** |
| --- | --- |
| btn-default | 默认样式 |
| btn-primary | 首选项 |
| btn-success | 成功样式 |
| btn-info | 一般信息 |
| btn-warning | 警告样式 |
| btn-danger | 危险样式 |
| btn-link | 链接样式 |

<div class="container">

<div class="row">

<button type="button" class="btn btn-default">（默认样式）Default</button>

<button type="button" class="btn btn-primary">（首选项）Primary</button>

<button type="button" class="btn btn-success">（成功）Success</button>

<button type="button" class="btn btn-info">（一般信息）Info</button>

<button type="button" class="btn btn-warning">（警告）Warning</button>

<button type="button" class="btn btn-danger">（危险）Danger</button>

<button type="button" class="btn btn-link">（链接）Link</button>

</div>

</div>

**3）不同尺寸的按钮**

* Bootstrap 为按钮提供了大、默认、小和很小几个尺寸。

<div class="container">

<div class="row">

<button type="button" class="btn btn-default btn-lg">（大按钮）Large button</button>

<button type="button" class="btn btn-default">（默认尺寸）Default button</button>

<button type="button" class="btn btn-default">（默认尺寸）Default button</button>

<button type="button" class="btn btn-default btn-xs">（超小尺寸）Extra small button</button>

</div>

</div>

* 通过给按钮添加 .btn-block 类可以将其拉伸至父元素100%的宽度，而且按钮也变为了块级（block）元素。

<div class="container">

<div class="row">

<button type="button" class="btn btn-default btn-lg btn-block">（块级元素）Block level button</button>

</div>

</div>

**4）按钮激活状态**

为按钮元素添加 .active 类，设置其为激活状态。当按钮处于激活状态时，其表现为被按压下去（底色更深、边框夜色更深、向内投射阴影）。

<div class="container">

<div class="row">

<button type="button" class="btn btn-default btn-lg active">Button</button>

<a href="#" class="btn btn-default btn-lg active" role="button">Link</a>

</div>

</div>

**5）按钮禁用状态**

通过为按钮的背景设置 opacity 属性就可以呈现出无法点击的效果。

* 为 <button> 元素添加 disabled 属性，使其表现出禁用状态。

<button type="button" class="btn btn-default btn-lg" disabled="disabled">Button</button>

**跨浏览器兼容性**

Internet Explorer 9 及更低版本的浏览器将会把按钮中的文本绘制为灰色，并带有恶心的阴影，目前还没有解决办法。

* 为基于 <a> 元素创建的按钮添加 .disabled 类。

<a href="#" class="btn btn-default btn-lg disabled" role="button">Link</a>

**5. 图片**

**1）响应式图片**

通过为图片添加 .img-responsive 类可以让图片支持响应式布局。

如果需要让使用了 .img-responsive 类的图片水平居中，请使用 .center-block 类，不要用 .text-center。

<div class="container">

<div class="row">

<img src="imgs/1.jpg" class="img-responsive" alt="Responsive image">

</div>

<div class="row">

<img src="imgs/1.jpg" class="img-responsive center-block" alt="Responsive image">

</div>

</div>

**2）图片形状**

通过为  元素添加以下相应的类，可以让图片呈现不同的形状。

| **类名** | **描述** |
| --- | --- |
| img-rounded | 圆角矩形 |
| img-circle | 圆形 |
| img-thumbnail | 矩形 |

IE 8 不支持 CSS3 中的圆角属性。

<div class="container">

<div class="row">

<img src="imgs/1.jpg" alt="..." class="img-rounded">

<img src="imgs/1.jpg" alt="..." class="img-circle">

<img src="imgs/1.jpg" alt="..." class="img-thumbnail">

</div>

</div>

**6. 文本**

**1）文本对齐**

通过以下文本对齐类，可以简单方便的将文字重新对齐。

| **类名** | **描述** |
| --- | --- |
| text-left | 左对齐 |
| text-center | 居中对齐 |
| text-right | 右对齐 |
| text-justify | 两端对齐，段落中超出屏幕部分文字自动换行 |
| text-nowrap | 段落中超出屏幕部分不换行 |

<div class="container">

<div class="row">

<p class="text-left">Left aligned text.</p>

<p class="text-center">Center aligned text.</p>

<p class="text-right">Right aligned text.</p>

<p class="text-justify">Justified text.</p>

<p class="text-nowrap">No wrap text.</p>

</div>

</div>

**2）改变大小写**

通过以下这几个类可以改变文本的大小写。

| **类名** | **描述** |
| --- | --- |
| text-lowercase | 小写 |
| text-uppercase | 大写 |
| text-capitalize | 首字母大写 |

<div class="container">

<div class="row">

<p class="text-lowercase">Lowercased text.</p>

<p class="text-uppercase">Uppercased text.</p>

<p class="text-capitalize">Capitalized text.</p>

</div>

</div>

**3）文本颜色**

Bootstrap 允许为文本设置不同颜色表示不同含义，具体如下：

| **类名** | **描述** |
| --- | --- |
| text-muted | 提示，使用浅灰色 |
| text-primary | 首选项，使用蓝色 |
| text-success | 成功，使用浅绿色 |
| text-info | 信息，使用浅蓝色 |
| text-warning | 警告，使用黄色 |
| text-danger | 危险，使用褐色 |

<div class="container">

<div class="row">

<p class="text-muted">提示信息</p>

<p class="text-primary">首选项</p>

<p class="text-success">成功</p>

<p class="text-info">信息</p>

<p class="text-warning">警告</p>

<p class="text-danger">危险</p>

</div>

</div>

**本笔记由 *金云龙* 编写**

作者博客地址：[龙哥有话说](http://www.longestory.com)

如转载或发布，请注明作者名称