**移动端WEB开发之响应式布局 – LUO JIN – 2020-07-14 (TUES)**

[1. 响应式开发 2](#_Toc45751754)

[1.1 响应式开发原理 2](#_Toc45751755)

[1.2 响应式布局容器 2](#_Toc45751756)

[2. Bootstrap前端开发框架 3](#_Toc45751757)

[2.1 Bootstrap 简介 3](#_Toc45751758)

[2.1 Bootstrap 简介 3](#_Toc45751759)

[2.2 Bootstrap 使用 3](#_Toc45751760)

[3. Bootstrap 栅格系统 5](#_Toc45751761)

[3.1 栅格系统简介 5](#_Toc45751762)

[3.2 栅格选项参数 5](#_Toc45751763)

[3.3 列嵌套 6](#_Toc45751764)

[3.4 列偏移 7](#_Toc45751765)

[3.5 列排序 8](#_Toc45751766)

[3.6 响应式工具 8](#_Toc45751767)

[4. 阿里百秀首页案例 【案例】：阿里百秀移动端首页 9](#_Toc45751768)

[5. 移动端布局总结 10](#_Toc45751769)

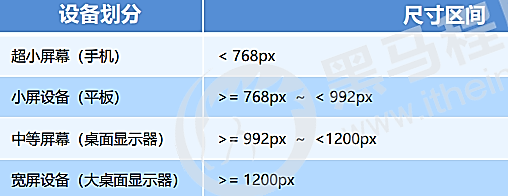
[5.1 移动端主流方案 10](#_Toc45751770)

[5.2移动端技术选型 10](#_Toc45751771)

# 1. 响应式开发

## 1.1 响应式开发原理

就是使用媒体查询针对**不同宽度的设备进行布局和样式的设**置，从而适配不同设备的目的。



## 1.2 响应式布局容器

响应式需要一个**父级**做为布局容器，来**配合子级元素**来实现变化效果。

**原理就是在不同屏幕下，通过媒体查询来改变这个布局容器的大小**，**再改变里面子元素的排列方式和大小，**从而实现不同屏幕下，看到不同的页面布局和样式变化。

**平时我们的响应式尺寸划分：**

1. 超小屏幕（手机，小于 768px）：设置宽度为 100%
2. 小屏幕（平板，大于等于 768px）：设置宽度为 750px
3. 中等屏幕（桌面显示器，大于等于 992px）：宽度设置为 970px
4. 大屏幕（大桌面显示器，大于等于 1200px）：宽度设置为 1170px

但是我们也可以根据实际情况自己定义划分

【案例】：响应式导航



【案例】：需求分析

① 当我们屏幕大于等于800像素，我们给nav宽度为800px，因为里面子盒子需要浮动，所以nav需要清除浮动。

② nav里面包含8个小li 盒子，每个盒子的宽度定为 100px， 高度为 30px，浮动一行显示。

③ **当我们屏幕缩放，宽度小于800像素**的时候， nav盒子宽度修改为 100% 宽度。

④ nav里面的8个小li，宽度修改为 33.33%，这样一行就只能显示3个小li ，剩余下行显示。

快速开发

# 2. Bootstrap前端开发框架

## 2.1 Bootstrap 简介

Bootstrap 来自 **Twitter（推特），**是目前最受欢迎的前端框架。Bootstrap 是基于 HTML、CSS 和 JAVASCRIPT 的，它简洁灵活，使得 Web 开发更加快捷。

中文官网：<http://www.bootcss.com/>

官网：<http://getbootstrap.com/>

推荐使用：<http://bootstrap.css88.com/>

**框架**：顾名思义就是一套架构，它有一套比较完整的网页功能解决方案，而且控制权在框架本身，**有预制样式库、组件和插件**。使用者要按照框架所规定的某种规范进行开发。

## 2.1 Bootstrap 简介

1. 优点

* 标准化的html+css编码规范
* 提供了一套简洁、直观、强悍的组件
* 有自己的生态圈，不断的更新迭代
* 让开发更简单，提高了开发的效率

2. 版本

2.x.x：停止维护,兼容性好,代码不够简洁，功能不够完善。

3.x.x：目前使用最多,稳定,但是放弃了 IE6-IE7。对 IE8 支持但是界面效果不好,偏向用于开发响应式布局、 **移动设备优先的 WEB 项目**。

4.x.x ：最新版，目前还不是很流行

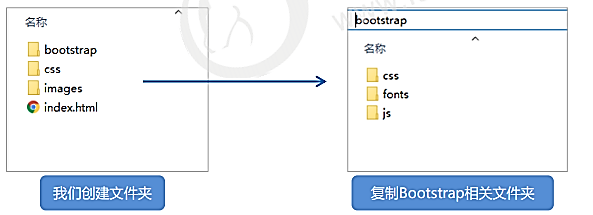
## 2.2 Bootstrap 使用

在现阶段我们还没有接触JS相关课程，所以我们只考虑使用它的样式库。

控制权在框架本身，使用者要按照框架所规定的某种规范进行开发。

Bootstrap 使用四步曲： **1. 创建文件夹结构** 2. 创建 html 骨架结构 3. 引入相关样式文件 4. 书写内容

1. 创建文件夹结构



1. 创建html骨架结构

|  |
| --- |
| <!--要求当前网页使用IE浏览器最高版本的内核来渲染-->  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <!--视口的设置：视口的宽度和设备一致，默认的缩放比例和PC端一致，用户不能自行缩放-->  <meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=0">  <!--[if lt IE 9]>  <!--解决ie9以下浏览器对html5新增标签的不识别，并导致CSS不起作用的问题-->  <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>  <!--解决ie9以下浏览器对 css3 Media Query 的不识别 -->  <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>  <![endif]--> |

1. 引入相关样式文件

|  |
| --- |
| <!-- Bootstrap 核心样式-->  <link rel="stylesheet" href="bootstrap/css/bootstrap.min.css"> |

1. 书写内容  
   直接拿Bootstrap 预先定义好的样式来使用

修改Bootstrap 原来的样式，注意权重问题

学好Bootstrap 的关键在于知道它定义了哪些样式，以及这些样式能实现什么样的效果

2.3 布局容器

Bootstrap **需要为页面内容和栅格系统包裹一个 .container 容器**，它提供了两个作此用处的类。

|  |  |
| --- | --- |
| **1. container 类**   响应式布局的容器 固定宽度   大屏 ( >=1200px) 宽度定为 1170px   中屏 ( >=992px) 宽度定为 970px   小屏 ( >=768px) 宽度定为 750px   超小屏 (100%) | **2. container-fluid 类**   流式布局容器 百分百宽度   占据全部视口（viewport）的容器。 |

# 3. Bootstrap 栅格系统

## 3.1 栅格系统简介

栅格系统英文为“**grid systems”**,也有人翻译为“网格系统”，它是指将页面布局划分为等宽的列，然后通过列数的定义来模块化页面布局

Bootstrap 提供了一套响应式、移动设备优先的流式栅格系统，随着屏幕或视口（viewport）尺寸的增加， 系统会自动分为最多12列

## 3.2 栅格选项参数

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。



* 按照不同屏幕划分为1~12 等份
* 行（row） 可以去除父容器作用15px的边距
* xs-extra small：超小； sm-small：小； md-medium：中等； lg-large：大；
* 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
* 每一列默认有左右15像素的 padding
* 可以同时为一列指定多个设备的类名，以便划分不同份数 例如 class="col-md-4 col-sm-6

<!-- 以IE最高版本来兼容 -->

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

     <!--[if lt IE 9]>

      <script src="https://cdn.jsdelivr.net/npm/html5shiv@3.7.3/dist/html5shiv.min.js"></script>

      <script src="https://cdn.jsdelivr.net/npm/respond.js@1.4.2/dest/respond.min.js"></script>

    <![endif]-->

    <!-- 引入bootstrap 样式文件 -->

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

<div class="container">

        <!-- row 必须放在container 里边 -->

        <div class="row">

            <!-- 一共12份，在大屏幕下 col-lg-3 ，每个盒子占3分，4个盒子\*3=12分

               在中等屏幕下，每个盒子占4份 col-md-4 =  每一行只能放3个

               在小屏幕下，每个盒子占6份 col-sm-6 = 每一行只能放2个

               超小屏幕下，每个盒子占12份 col-xs-12 = 每行只能放1个

            -->

            <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">1</div>

            <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">2</div>

            <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">3</div>

            <div class="col-lg-3 col-md-4 col-sm-6 col-xs-12">4</div>

        </div>

如果孩子的分数相加等于12， 则孩子沾满整个container的宽度

如果孩子的分数相加小于12，？？会少一份，占不满整个container的宽度

如果孩子的分数相加大于12，？？会另起一行显示

## 3.3 列嵌套

栅格系统**内置的栅格系统将内容再次嵌套**。简单理解就是**一个列内再分成若干份小列**。我们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。



|  |
| --- |
| <!-- 列嵌套 -->  <div class="**col-sm-4**">  <div class="row">  <div class="**col-sm-6**">小列</div>  <div class="col-sm-6">小列</div>  </div>  </div> |
| <div class="container">          <!-- row 必须放在container 里边 -->          <div class="row">              <!-- 在中等屏幕下，划分成3等分 -->              <div class="col-md-4">                  <!-- 列的嵌套最好加一个row, 消除父元素的padding值，而且高度自动和父级一样高 -->                  <div class="row">                      <div class="col-md-6">a</div>                      <div class="col-md-6">b</div>                  </div>              </div>              <div class="col-md-4">2</div>              <div class="col-md-4">3</div>          </div>      </div> |

## 3.4 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）。



|  |
| --- |
| <!-- 列偏移 -->  <div class="row">  <div class="**col-lg-4**">1</div>  <div class="**col-lg-4 col-lg-offset-4**">2</div>  </div> |
| <style>          .row>div{              height: 50px;              background-color: cornflowerblue;          }      </style>  </head>  <body>      <div class="container">          <!-- row 必须放在container 里边 -->          <div class="row">              <!-- 总12份：左边4份， 【中间4份】，右边4份，右边向右偏移4份 col-md-offset-4 -->              <div class="col-md-3">left</div>              <div class="col-md-3 col-md-offset-6">right</div>          </div>          <div class="row">              <!-- 如果只有一个盒子，那么就偏移4/2=2 -->              <div class="col-md-8 col-md-offset-2">中间盒子</div>          </div>      </div> |

## 3.5 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。



|  |
| --- |
| <!-- 列排序 -->  <div class="row">  <div class="**col-lg-4 col-lg-push-8**">左侧</div>  <div class="**col-lg-8 col-lg-pull-4**">右侧</div>  </div> |

## 3.6 响应式工具

为了加快**对移动设备友好的页面开**发工作，利用媒体查询功能，并使用这些工具类可以方便的针对不同设备 展示或隐藏页面内容。 Bootstrap 其他（按钮、表单、表格） 请参考Bootstrap 文档



与之相反的是，visible-xs visible-sm visible-md visible-lg 显示某个页面的内容

# 4. 阿里百秀首页案例 【案例】：阿里百秀移动端首页

**技术选型**

* 方案：我们采取响应式页面开发方案
* 技术：bootstrap框架
* 设计图： 本设计图采用 1280px 设计尺寸

【案例】：需求分析

1. 页面布局分析
2. 屏幕划分分析

① 屏幕缩放发现 中屏幕 和 大屏幕布局 是一致的。 因此我们**列** **定义为 col-md**- 就可以了， md 是大于等于 970 以上的

② 屏幕缩放发现 小屏幕 布局发生变化，因此我们需要为 小屏幕根据需求改变布局

③ 屏幕缩放发现 超小屏幕布局又发生变化，因此我们需要为 超小屏幕根据需求改变布局

④ 策略： **我们先布局 md以上的 pc端布局**，最后根据实际需求在修改 小屏幕 和 超小屏幕的 特殊布局样式

【案例】：页面制作

Bootstrap 使用四步曲： **1. 创建文件夹结构** 2. 创建 html 骨架结构 3. 引入相关样式文件 4. 书写内容



container 宽度修改

**因为本效果图采取 1280的宽度， 而Bootstrap 里面 container宽度 最大为 1170px**，因此我们需要手动改下container宽度

|  |
| --- |
| **/\* 利用媒体查询修改 container宽度适合效果图宽度 \*/**  @media (min-width: 1280px) { .container { width: 1280px; } } |

# 5. 移动端布局总结

## 5.1 移动端主流方案

|  |  |
| --- | --- |
| **1. 单独制作移动端页面（主流）**  京东商城手机版  淘宝触屏版  苏宁易购手机版 .... | **2. 响应式页面兼容移动端（其次）**  三星手机官网 .... |

## 5.2移动端技术选型

* 流式布局（百分比布局）
* **flex 弹性布局（推荐）**
* **rem适配布局（推荐）**
* 响应式布局

建议： 我们选取一种主要技术选型， 其他技术做为辅助，这种混合技术开发