**（〇）目录结构**

bootstrap.html

jq.js

normalize.css

├── css/

│ ├── bootstrap.css

│ ├── bootstrap.min.css

├── js/

│ ├── bootstrap.js

│ ├── bootstrap.min.js

└── img/

├── glyphicons-halflings.png

└── glyphicons-halflings-white.png

**（一）一个标准的Bootstrap模板需要的内容**

<!DOCTYPE html>  
<**html**>  
<**head**>  
 <**title**></**title**>  
 <**meta** name="viewport" content="width=device-width, initial-scale=1.0">  
 <!-- Bootstrap -->  
 <**link** href="css/bootstrap.min.css" rel="stylesheet" media="screen">  
</**head**>  
<**body**>  
<**h1**>Hello, world!</**h1**>  
<**script** src="http://code.jquery.com/jquery.js"></**script**>  
<**script** src="js/bootstrap.min.js"></**script**>  
</**body**>  
</**html**>

简单来说，

①需要<!DOCTYPE html>这个h5文档，

②需要加载一个bootstrap的css，一个jquery，一个bootstrap的js（按顺序）

③jquery要放在前面，bootstrap要放在文档的最后。

下载bootstrap（来源于官方）：

编译好的：（一般下这个）

<http://getbootstrap.com/2.3.2/assets/bootstrap.zip>

源码：

<https://github.com/twbs/bootstrap/archive/v2.3.2.zip>

**（二）重置样式表和启用响应式**

①重置：

简单来说，目的是为了统一样式在不同浏览器上的显示。

下载链接：

<https://necolas.github.io/normalize.css/4.1.1/normalize.css>

来源于这个：

<http://necolas.github.io/normalize.css/>

②启用响应式

在普通Bootstrap页面结构的基础上，添加代码（最好是在Bootstrap原有的css之后）：

<**link** href="css/bootstrap-responsive.css" rel="stylesheet">

**（三）栅格系统**

①简单来说，bootstrap利用栅格系统，形成一个12列，940px宽的容器。

通过这个12列的容器，可以形成类似表格一样的布局（利用display: table）这个css属性。

好处是对齐，有序排列会很简单。

例如：

整体布局是左边栏+右上右中单页面；

左定宽满高，右部撑满宽度；

右上定高满宽；

右中撑满剩余区域；

右中三行布局；

第一行4列，等宽；

第二行2列，宽度比1：3；

第三行1列；

每行全部撑满。

（但右中上下方向不需要撑满，这个没研究到）。

如代码：（后面解释）

<!DOCTYPE html>  
<**html**>  
<**head**>  
 <**title**>Bootstrap 101 Template</**title**>  
 <**link** href="normalize.css" rel="stylesheet" media="screen">  
 <**meta** name="viewport" content="width=device-width, initial-scale=1.0">  
 <!-- Bootstrap -->  
 <**link** href="css/bootstrap.min.css" rel="stylesheet" media="screen">  
 <**link** href="css/bootstrap-responsive.css" rel="stylesheet">  
</**head**>  
<**body**>  
<**style**>  
 .test {  
 }  
  
 .box {  
 height: 150px;  
 }  
  
 .show-grid [class\*="span"] {  
 background-color: #eee;  
 text-align: center;  
 -webkit-border-radius: 3px;  
 -moz-border-radius: 3px;  
 border-radius: 3px;  
 min-height: 40px;  
 line-height: 40px;  
 }  
  
 .show-grid [class\*="span"]:hover {  
 background-color: #ddd;  
 }  
  
 .body {  
 position: absolute;  
 width: 100%;  
 height: 100%;  
 overflow: hidden;  
 }  
  
 .my-nav {  
 position: absolute;  
 left: 0;  
 top: 0;  
 bottom: 0;  
 width: 300px;  
 border-right: 1px solid red;  
 }  
  
 .head {  
 position: absolute;  
 left: 301px;  
 right: 0;  
 top: 0;  
 height: 150px;  
 border-bottom: 1px solid yellowgreen;  
 }  
  
 .main {  
 position: absolute;  
 left: 301px;  
 right: 0;  
 top: 151px;  
 bottom: 0;  
 overflow: auto;  
 }  
</**style**>  
<**div** class="body">  
 <**nav** class="my-nav"></**nav**>  
 <**header** class="head"></**header**>  
 <**main** class="main">  
 <**div** class="row-fluid test show-grid">  
 <**div** class="span3 box">  
 A  
 </**div**>  
 <**div** class="span3 box">  
 A  
 </**div**>  
 <**div** class="span3 box">  
 A  
 </**div**>  
 <**div** class="span3 box">  
 A  
 </**div**>  
 </**div**>  
 <**div** class="row-fluid test show-grid">  
 <**div** class="span3 box">  
 B1  
 </**div**>  
 <**div** class="span9 box">  
 B2  
 </**div**>  
 </**div**>  
 <**div** class="row-fluid test show-grid">  
 <**div** class="span12 box">  
 C  
 </**div**>  
 </**div**>  
 </**main**>  
  
</**div**>  
<**script** src="./jq.js"></**script**>  
<**script** src="js/bootstrap.min.js"></**script**>  
</**body**>  
</**html**>

②spanX

栅格系统是12列宽度，class为span开头，后面跟数字时，数字为几，其就代表几列宽度。

例如：class="span12"表示12列宽度的一个DOM，class="span4"表示4列宽度。

3个4列宽度的DOM其总宽度和span12是相同的。

给一个官方的示例图：

数字表示span的类名

③栅格的超限

假如子栅格总宽度超过12，那么就会错行，并且从第二个栅格开始，前面会间隔。

总之超限是不对的。

**（四）响应式和非响应式**

①简单来说，响应式布局，会自动撑满父div，而非响应式则不会撑满。

默认是非响应式布局；

②非响应式就是普通的栅格系统咯；

③对于栅格系统来说，响应式（或者成为流式布局）就是在类名后面加“-fluid"这样。

例如：

<**div** class="**row-fluid**">

④最大宽度

非响应式布局最大宽度是1170px（前提是窗口的可视宽度超过1200px）；

响应式的最大宽度是100%，即可以铺满布局；

⑤最小宽度

在浏览器可视窗口低于767px的时候，宽度自动转为100%，并在垂直方向堆叠。——无论是否是流式（响应式）布局。

⑥高度

在spanX类的父DOM不使用row或者row-fluid的情况下，父div是没有高度的（即使spanX有设置高度）；

但父DOM的父DOM是有高度的，例如：

<**div** class="body">  
 <**div** class="show-grid">  
 <**div** class="span3">  
 A  
 </**div**>  
 <**div** class="span6 offset3">  
 A  
 </**div**>  
 </**div**>  
</**div**>

最外层的class="body"的有高度；

次外层的class="show-grid"只有宽度没有高度；

内层正常。

**（五）栅格的嵌套**

①简单来说，就是在栅格系统里，再套栅格，形成树形结构；

②非响应式的嵌套和响应式的嵌套不同；

③非响应式的嵌套：

栅格总共是12；

假如子栅格是6，

那么孙栅格总共宽度不能超过6；（例如2+3，或者1+5之类）

④非响应式嵌套超出范围：

假如超过了，该子栅格的所有孙栅格的宽度则变为100%（相对于子栅格）；（例如3+4，或者5+6之类）；

若某个孙栅格的宽度大于子栅格（例如子4孙5），则孙栅格会超出子栅格的范围；

在上一种情况下，如果子1宽度6，子1的孙宽度为10，子2宽度6，那么子1的孙栅格的宽度虽然超范围，但并不会挤压子2的宽度（只会覆盖子1和子2中口原有的空余区域）；

如：

<**div** class="row show-grid">  
 <**div** class="span6">  
 <**div** class="row">  
 <**div** class="span9">A</**div**>  
 </**div**>  
 </**div**>  
 <**div** class="span6">C</**div**>

</**div**>

效果如图：

A只是和C紧密挨着而已



⑤响应式嵌套：

孙栅格的总宽度要为12（无论子栅格是多少）；

相当于每个子栅格都是一个父栅格

如代码：

<**div** class="body">  
 <**div** class="row-fluid show-grid">  
 <**div** class="span6">  
 <**div** class="row">  
 <**div** class="span6">A</**div**>  
 </**div**>  
 </**div**>  
 <**div** class="span6">C</**div**>  
 </**div**>  
</**div**>

效果如图：



**（六）栅格偏移offset**

①简单来说，在原有位置基础上，向右偏移几列

②offsetX 偏移X列

基于原本位置，偏移X列。

例如：

<**div** class="body">  
 <**div** class="row show-grid">  
 <**div** class="span3">  
 A  
 </**div**>  
 <**div** class="span3 offset3">  
 A  
 </**div**>  
 </**div**>  
 <**div** class="row show-grid">  
 <**div** class="span3">  
 A  
 </**div**>  
 <**div** class="span6 ">  
 A  
 </**div**>  
 </**div**>  
</**div**>

两行的左、右边是对齐的。

**（七）固定布局container**

①简单来说，固定布局是自动居中，自动自适应，适合只有一块的页面。

②可以在其中使用栅格布局，

例如：

<**div** class="body container">  
 <**div** class="row show-grid">  
 <**div** class="span2">  
 A  
 </**div**>  
 <**div** class="span10">  
 B  
 </**div**>  
 </**div**>  
</**div**>

则是左2右10。

**（八）流式布局container-fluid**

①简单来说，流式布局和普通布局的区别在于，流式布局会自适应（100%width）；

也可以直接充当栅格父容器来用——虽然我觉得这样并不好。

②流式布局最好配合流式栅格使用。

如代码：

<**div** class="body container-fluid show-grid">  
 <**div** class="row-fluid">  
 <**div** class="span2">  
 A  
 </**div**>  
 <**div** class="span10">  
 B  
 </**div**>  
 </**div**>  
</**div**>

效果是

**（九）排版**

①原文链接：

<http://v2.bootcss.com/base-css.html>

个人感觉有些标签是html自带，只不过bootstrap给其加了一些样式。

以下只总结特殊的

②文字对齐：（类名）

左对齐：text-left

居中：text-center

右对齐：text-right

③强调（颜色）：（类名）

变淡：muted

黄色：text-warning

红色：text-error

蓝色：text-info

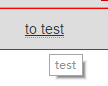
绿色：text-success

④缩略语：（标签）

abbr

缩略内容放在title属性中

<**abbr** title="test">to test</**abbr**>



类名：.initialism 加到abbr标签，让title的字体更小一些

⑤地址：（标签）

address

有一些基本的样式，但并没有什么特殊的。

⑥引用：（标签）

默认引用：blockquote

命名来源：small

来源名称：cite

右对齐（类名）：pull-right

⑦列表：（标签）

无序：ul

有序：ol

列表项：li

无样式列表（类名）：unstyled（只对直接子节点有效）

行内列表（类名）：inline（效果是让列表同行显示，并有间隔）；

⑧描述：（标签）

父：dl

着重/标题：dt

普通：dd

水平描述（类名，加到父）：dl-horizontal（效果是着重右对齐，普通左对齐，着重和描述对应的开始行在同一行）

⑨代码（标签）：

行内代码：code（类似<>尖括号需要用转义字符，左是&lt，右是&gt）；

基本代码块：pre（同样转义）；

把该区域设置成最大高度为350px并带有一个Y轴滚动条（类名）：pre-scrollable

⑩表格（类名）

横线分割行：.table

以上基础上，下同：（加在table标签）

【1】标题加黑：thead，然后是tr，具体每一格用th（而不是td）

<**table** class="table">  
 <**thead**>  
 <**tr**>  
 <**th**>id</**th**>  
 <**th**>name</**th**>  
 </**tr**>  
 </**thead**>  
 <**tbody**>  
 <**tr**>  
 <**td**>1</**td**>  
 <**td**>wang</**td**>  
 </**tr**>  
 </**tbody**>  
 <**tfoot**>  
 <**tr**>  
 <**td**>total</**td**>  
 <**td**>names</**td**>  
 </**tr**>  
 </**tfoot**>  
</**table**>

【2】tbody内部，斑马样式，奇数行加重颜色：table-striped

【3】表格边框：table-bordered

【4】鼠标悬停样式：table-hover

【5】表格内部更紧凑（行高更少）：table-condensed

行属性（类名，背景颜色，加在tr标签上）：

变绿：success

红：error

黄：warning

蓝：info