# 复习

变形：一个元素可以同时设置多个变形，用空格隔开

transform: rotate(10deg) scale(1) skew(10deg, 20deg) translate(100px,100px);

定义动画：

|  |
| --- |
| 1. @-webkit-keyframes 动画名称 { 2. 0% {} 3. n% {} 4. 100% {} 5. } |

每一个状态必须一一对应

调用：

|  |
| --- |
| 1. -webkit-animation: 动画名称 一次动画完成时间s 缓冲描述 延迟时间s 动画次数infinite 2. 是否补全反方向动画alternate 是否保持最后一帧forwards; |

百分比布局：

标准流和浮动：

1width，padding，margin父盒子内容宽比值

2 height父盒子内容高的比值

3 border不能使用百分数

# 百分比布局

## 内减盒模型

盒子设置width,height，再设置padding，border盒子的占有区域变大，内容区域不变。这种称为外扩盒模型

css3增加的属性，内减盒模型

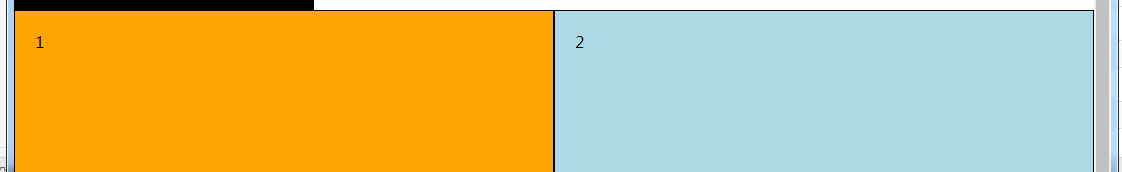
盒子设置width,height，再设置padding，border盒子的占有区域不变，内容区域变小。

width:盒子的占有宽度

height:盒子的占有高度

box-sizing: border-box;

|  |
| --- |
| 1. .box2 { 2. /\*通栏\*/ 3. width: 100%; 4. height: 200px; 5. background-color: #eee; 6. } 7. .box2 p { 8. float: left; 9. /\*width:占有宽度 一定是内减盒模型\*/ 10. width: 50%; 11. height: 100%; 12. background-color: orange; 13. padding: 20px; 14. border: 1px solid #000; 15. box-sizing: border-box; 16. } 17. .box2 p:last-child { 18. background-color: lightblue; 19. } |



## 弹性盒

父盒子设置弹性盒容器，子盒子自动并排显示不需要再设置浮动。

子盒子利用弹性盒均分父盒子宽度。

弹性盒容器：

|  |
| --- |
| 1. display: -webkit-flex; |

|  |
| --- |
| 1. 子盒子均分：-webkit-flex: 数值；表示均分父盒子宽度的份数 |

## 固比固模型

第1部分和3部分宽度是固定值，第2部分宽度自适应。

* 浮动

|  |
| --- |
| 1. .box p { 2. float: left; 3. height: 200px; 4. } 5. .box p:first-child { 6. width: 200px; 7. background-color: green; 8. } 9. .box p:last-child { 10. width: 200px; 11. background-color: green; 12. } 13. /\*2号元素宽度自适应 计算\*/ 14. .box p:nth-child(2) { 15. /\*计算\*/ 16. width: calc(100% - 400px); 17. height: 200px; 18. background-color: orange; 19. } |

* 弹性盒

|  |
| --- |
| 1. .box { 2. /\*父盒子设置弹性盒容器\*/ 3. width: 100%; 4. display: -webkit-flex; 5. /\*子盒子自动并排显示 不需要设置浮动\*/ 6. } 7. .box p:first-child { 8. width: 200px; 9. height: 200px; 10. background-color: lightblue; 11. } 12. .box p:nth-child(2) { 13. /\*宽度自适应\*/ 14. -webkit-flex: 2222; 15. height: 200px; 16. background-color: green; 17. } 18. .box p:last-child { 19. width: 200px; 20. height: 200px; 21. background-color: orange; 22. } |

* 绝对定位（推荐）

|  |
| --- |
| 1. .box { 2. /\*width:占有宽度\*/ 3. width: 100%; 4. height: 200px; 5. background: #eee; 6. /\*利用padding挤出1,3位置\*/ 7. padding: 0 200px; 8. /\*内减盒模型\*/ 9. box-sizing: border-box; 10. position: relative; 11. } 12. .box p:first-child { 13. /\*绝对定位\*/ 14. position: absolute; 15. width: 200px; 16. height: 200px; 17. background-color: green; 18. left: 0; 19. } 20. .box p:last-child { 21. /\*绝对定位\*/ 22. position: absolute; 23. width: 200px; 24. height: 200px; 25. background-color: green; 26. right: 0; 27. top: 0; 28. } 29. .box p:nth-child(2) { 30. /\*标准流 父盒子内容宽的比值\*/ 31. width: 100%; 32. height: 200px; 33. background-color: orange; 34. } |

# 二、响应式网页制作

## 2.1 概念

同一个网页根据视口的不同，显示不同的版式。

响应式网页制作：responsive web design (RWD)

|  |  |
| --- | --- |
| 淘宝视口：1190px | 淘宝视口小于：1190px |
|  |  |

## 2.2 优缺点

优点：对于不同视口都可以显示非常饱满的网页结构，没有横向滚动条。

缺点：制作复杂，同时对于移动端而言，需要加载非常多的pc端的样式和图片等资源，影响加载速度，同时费流量。国内流量非常贵。（这也是国内纯响应式网站不多的原因。）

## 2.3 媒体查询

css3不需要借助js也可以实现媒体查询。

同一个html网页根据视口不同，显示不同的样式（加载不同的css）

外链式实现媒体查询：

screen:屏幕

media:媒体

min-width: 最小值，视口大于或者等于该值加载这个css

max-width:最大值，视口小于或者等于该值加载这个css

|  |
| --- |
| 1. <!-- 视口大于或者等于1200 p均分 01.css--> 2. <link rel="stylesheet" type="text/css" href="css/01.css" media="screen and (min-width: 1200px)"> 3. <!-- 视口小于1200px p独占一行 --> 4. <link rel="stylesheet" type="text/css" href="css/02.css" media="screen and (max-width: 1199px)"> |

媒体查询可以划分多个段口：使用and链接

|  |
| --- |
| 1. <!-- 视口大于或者等于1200 p均分 01.css--> 2. <link rel="stylesheet" type="text/css" href="css/01.css" media="screen and (min-width: 1200px)"> 3. <!-- 视口小于1200px 同时大于 700px p独占一行 --> 4. <link rel="stylesheet" type="text/css" href="css/02.css" media="screen and (max-width: 1199px) and (min-width: 700px)"> 5. <!-- 视口小于700 p改变颜色 加载03.css--> 6. <link rel="stylesheet" type="text/css" href="css/03.css" media="screen and (max-width: 699px)"> |

IE8不认识media，“留活口”

|  |
| --- |
| 1. <!-- 留活口 所有视口都会加载该css --> 2. <!-- 大于等于1200px加载01.css --> 3. <link rel="stylesheet" type="text/css" href="css/01.css"> 4. <!-- 小于1200px加载04.css --> 5. <link rel="stylesheet" type="text/css" href="css/04.css" media="screen and (max-width: 1199px)"> |

## 2.4 内嵌式实现媒体查询

|  |
| --- |
| 1. <style type="text/css"> 2. /\*清除默认样式\*/ 3. \* { 4. padding: 0; 5. margin: 0; 6. } 7. ul, ol { 8. list-style: none; 9. } 10. a { 11. text-decoration: none; 12. color: #000; 13. } 14. .box { 15. width: 100%; 16. overflow: hidden; 17. } 18. /\*媒体查询 and连接视口\*/ 19. @media screen and (min-width: 1200px) { 20. p { 21. float: left; 22. width: 50%; 23. height: 200px; 24. background-color: green; 25. } 26. .no2 { 27. background-color: orange; 28. } 29. } 30. /\*小于1200px 更改样式\*/ 31. @media screen and (max-width: 1199px) { 32. .box p { 33. width: 100%; 34. height: 200px; 35. background-color: red; 36. } 37. .box .no2 { 38. background-color: lightblue; 39. } 40. } 41. </style> |

# 三、bootstrap

## 3.1 bootstrap是什么

Bootstrap，来自 Twitter，是目前很受欢迎的前端框架。Bootstrap 是基于 HTML、CSS、JAVASCRIPT 的，它简洁灵活，使 Web 开发更加快捷。

它对HTML、CSS和JAVASCRIPT进行了封装，使它们使用起来更方便。我们只需要使用它已经设定好的类，或规则，即可快速应用它提供的功能。

Bootstrap 使用了一些 HTML5 元素和 CSS3 属性(如媒体查询)。为了让这些正常工作，需要使用 HTML5 文档类型<!DOCTYPE html>。如果在

Bootstrap 创建的网页开头不使用 HTML5 的文档类型（Doctype），可能会面临一些浏览器显示不一致的问题。

## 3.2 怎么使用bootstrap

1.下载

http://www.bootcss.com/ 点击bootstrap3中文文档，点击下载bootstrap3

2.下载“用于生产环境的”

下载完后它有三个文件夹css,fonts,js

3.在html文档中引入bootstrap

a) bootstrap.min.css

b) bootstrap.min.js

c) jquery

引入css中的“bootstrap.min.css”和js文件夹下的bootstrap.min.js。由于bootstrap是基于jquery的，所以在引入”bootstrap.min.js”前，要先引入jquery。并且js和jquery要放在body的最底部。

4. 使用html5的文档类型声明

<!doctype html>，bootstrap用到了h5，所以它包含一个html5版本的DOCTYPE

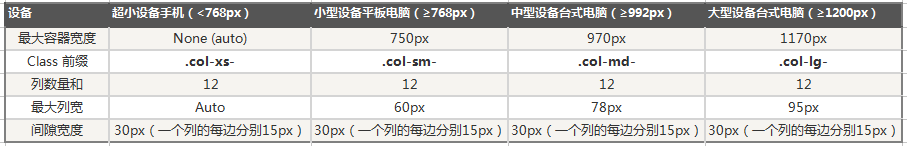
5. 添加适用于移动端的meta标签

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

因为现在越来越多的用户使用移动设备,为了让 Bootstrap 开发的网站对移动设备友好，确保适当的绘制和触屏缩放，需要在网页的 head 之中添加 viewport meta 标签

## 3.3 栅格系统

不同设备参数说明：



1. 行必须放置在 .container class 内，以便获得bootstrap设置的对齐（alignment）和内边距（padding）

2. 使用.row来创建一行，每行又分为多个列。

3. 内容应该放置在列内，且唯有列可以是行的直接子元素。row>col 正确写法 行>列

4. 预定义的网格类，比如 .row 和 .col-md-4，可用于快速创建网格布局。

栅格系统抽象模板：

|  |
| --- |
| 1. <div class="container"> 2. <div class="row"> 3. <div class="col-md-">列</div> 4. <div class="col-md-">列</div> 5. <div class="col-md-">列</div> 6. </div> 7. <div class="row"> 8. <div class="col-md-">列</div> 9. <div class="col-md-">列</div> 10. <div class="col-md-">列</div> 11. </div> 12. </div> |

## 3.4排版

使用 Bootstrap 的排版特性，您可以创建标题、段落、列表及其他内联元素，实际上它是把大部分在HTML的基本标签加了样式。所以这部分相对比较简单。

【标题】

h1-h6重新定义样式

【对齐类】

简单方便将文字对齐的类，它们一般应用在p标签上（用在div上不符合规范，用在span标签则不能生效）。

|  |
| --- |
| 1. <!-- 对齐类，加在p标签 --> 2. <p class="text-left">我是居左</p> 3. <p class="text-center">我是居中</p> 4. <p class="text-right">我是居右</p> |

【强调类】

这些强调类，通过颜色来表示强调。一般是给P元素加这些类名，会显示不同的颜色。

|  |
| --- |
| 1. <!-- 强调类 --> 2. <p class="text-muted">我是muted</p> 3. <p class="text-warning">我是warning</p> 4. <p class="text-info">info</p> 5. <p class="text-danger">danger</p> 6. <p class="text-success">success</p> |

## 3.5 表格

* 默认样式

.table

* 可选样式

.table-striped 为表格提供了斑马线的样式

.table-bordered 为表格增加边框(border)

.table-hover 为表格中的每一行赋予鼠标悬停样式。鼠标划过后会添加一个背景色。

.table-condensed 每个单元格的内补(padding)减半，可使表格更紧凑。

* 状态类通过这些状态类可以为行或单元格设置颜色。

.active 鼠标悬停在行或单元格上时所设置的颜色

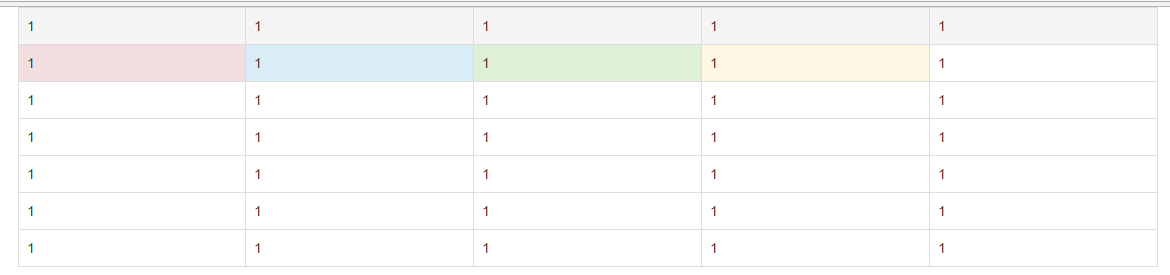
.success 标识成功或积极的动作

.info 标识普通的提示信息或动作

.warning 标识警告或需要用户注意

.danger 标识危险或潜在的带来负面影响的动作

|  |
| --- |
| 1. <table class="table table-bordered"> 2. <tr class="active"> 3. <td>1</td> 4. <td>1</td> 5. <td>1</td> 6. <td>1</td> 7. <td>1</td> 8. </tr> 9. <tr> 10. <td class="danger">1</td> 11. <td class="info">1</td> 12. <td class="success">1</td> 13. <td class="warning">1</td> 14. <td>1</td> 15. </tr> 16. <tr> 17. <td>1</td> 18. <td>1</td> 19. <td>1</td> 20. <td>1</td> 21. <td>1</td> 22. </tr> 23. <tr> 24. <td>1</td> 25. <td>1</td> 26. <td>1</td> 27. <td>1</td> 28. <td>1</td> 29. </tr> 30. <tr> 31. <td>1</td> 32. <td>1</td> 33. <td>1</td> 34. <td>1</td> 35. <td>1</td> 36. </tr> 37. <tr> 38. <td>1</td> 39. <td>1</td> 40. <td>1</td> 41. <td>1</td> 42. <td>1</td> 43. </tr> 44. <tr> 45. <td>1</td> 46. <td>1</td> 47. <td>1</td> 48. <td>1</td> 49. <td>1</td> 50. </tr> 51. </table> |



## 3.5 按钮

下面讲bootstrap中可作为按钮使用的标签或元素。为 <a>、<button> 或 <input> 元素添加按钮类（button class）即可使用 Bootstrap 提供的样式。

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

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

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

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

尽可能使用 <button> 元素来获得在各个浏览器上获得相同的显示效果

预定义样式

使用下面列出的类可以快速创建一个带有预定义样式的按钮。

<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>

按钮的尺寸

当你需要让按钮具有不同尺寸的时候，同样是使用 .btn-lg、.btn-sm 或 .btn-xs 就可以获得不同尺寸的按钮。

## 3.6字体图标

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。

不要和其他组件混合使用（应该创建一个嵌套的 <span> 标签，并将图标类应用到这个 <span> 标签上）

只对内容为空的元素起作用（图标类只能应用在不包含任何文本内容或子元素的元素上）

实例：图标字体

|  |
| --- |
| 1. <button type="button" class="btn btn-default"> 2. <span class="glyphicon glyphicon-align-left"></span> 3. </button> |

## 3.7 导航

Bootstrap 中的导航组件都依赖同一个 .nav 类，状态类也是共用的。改变修饰类可以改变样式。

标签页实例

注意 .nav-tabs 类依赖 .nav 基类

|  |
| --- |
| <ul class="nav nav-tabs">  <li role="presentation" class="active"><a href="#">Home</a></li>  <li role="presentation"><a href="#">Profile</a></li>  <li role="presentation"><a href="#">Messages</a></li>  </ul> |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |