# 移动web和响应式

第2天课堂笔记（本课程共4天）

目录

[移动web和响应式 1](#_Toc478168770)

[一、视口复习 2](#_Toc478168771)

[二、手机浏览器内核 3](#_Toc478168772)

[三、百分比布局 5](#_Toc478168773)

[3.1 相对于谁 5](#_Toc478168774)

[3.2 CSS3弹性盒属性display:-webkit-box; 6](#_Toc478168775)

[3.3 CSS3新的盒模型box-sizing:border-box 7](#_Toc478168776)

[3.4 固比固 8](#_Toc478168777)

[3.5 min-width、max-width 9](#_Toc478168778)

[四、高度等比变化 10](#_Toc478168779)

[4.1 图片自动就是等比例变化的 10](#_Toc478168780)

[4.2 padding-top猫腻 10](#_Toc478168781)

[4.3 js 11](#_Toc478168782)

[4.4 display:table-cell 11](#_Toc478168783)

[五、em和rem 12](#_Toc478168784)

[六、媒体查询 15](#_Toc478168785)

[七、携程建设作业 16](#_Toc478168786)

# 一、视口复习

50英寸的电视机，分辨率是3840\*2160；

5.5英寸的小手机，分辨率是1920\*1080； PPI：401

PPI：pixal per inch 每平方英寸像素的个数。

手机的PPI很大，所以屏幕小但是像素多，所以颗粒感小，视网膜屏，让人看不到屏幕上的像素点。

如果用1:1的视口呈递网页，那么将会像从高空俯瞰整个页面。你想啊，你的手机的高度就是1920px，我们做的博雅互动总高才1415px。也就是说，如果用手机1:1的视口呈递页面，所有的页面都是高空俯瞰。文字很小，不可读。所以手机不可能用1:1的视口呈递网页。

人为规定了两个视口：

1）默认视口，基本都是980px，这是绝大多数PC版网页的版心宽度，就能让我们的手机正好卡主版心。这是一种妥协：各个网站还没有来得及做手机版网页的时候，就只能让手机用户用手机看PC版网页，视口980恰到好处；

2）约束后的视口，通过加上一个简单的meta标签，就能让手机使用约束后的视口来呈递网页：

|  |
| --- |
| <meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0" /> |

约数后的视口宽度都是在320~480之间（手机竖直使用的时候）

这个视口的尺寸，是手机厂商设置的，能够保证我们的文字比如16px，在自己的这个视口下清晰、大小刚刚合适。所以大屏幕的手机的约束视口 > 小屏幕手机的约束视口。

这就能够保证我们的网页可以用px写字号、写行高。比如我们的盒子尺寸是200px\*200px，指的是用视口的像素点。在手机呈递的时候，屏幕的真实物理像素点，可能是635px\*635px个像素点；别的手机呢，真实物理像素点可能是595px\*595px。绝对尺寸不一样，给用户在手机屏幕上的感觉是一致的。

最最重要的一句话：前端开发工程师，丝毫不关心手机的分辨率，我们只关心视口。

# 二、手机浏览器内核

在移动端，仅有四个独立的浏览器内核，分别为微软的Trident、火狐的Gecko、开源内核Webkit、Opera的Presto。

目前微软的Trident在移动终端上主要为WP7、8系统内置浏览器。Opera的Presto内核主要为 Opera Mobile、OperaMini、欧朋浏览器以及欧朋HD Beta版。**Webkit内核的适用范围则较为广泛，Android原生浏览器、苹果的Safari、谷歌的Chrome(Android4.0使用)都是基于Webkit开源内核开发的**。

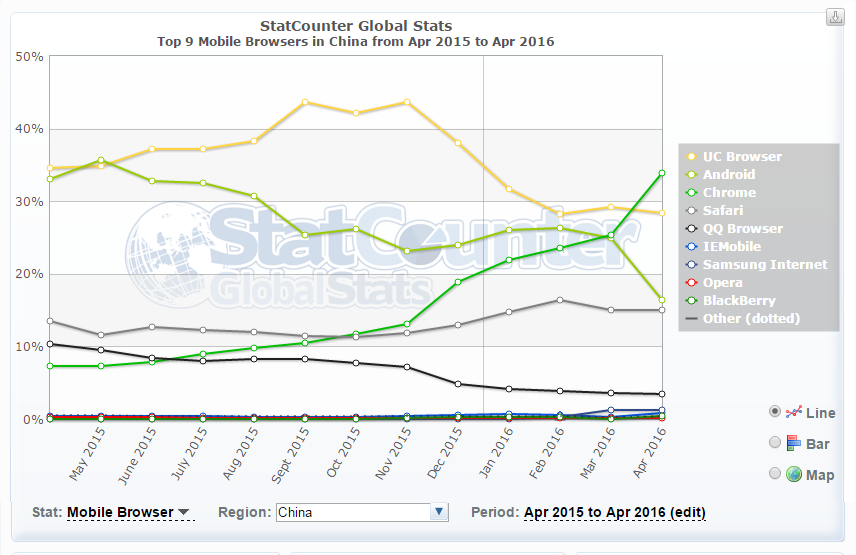
兼容的前缀：

|  |
| --- |
| 1. -ms- 2. -moz- 3. -o- 4. -webkit- |

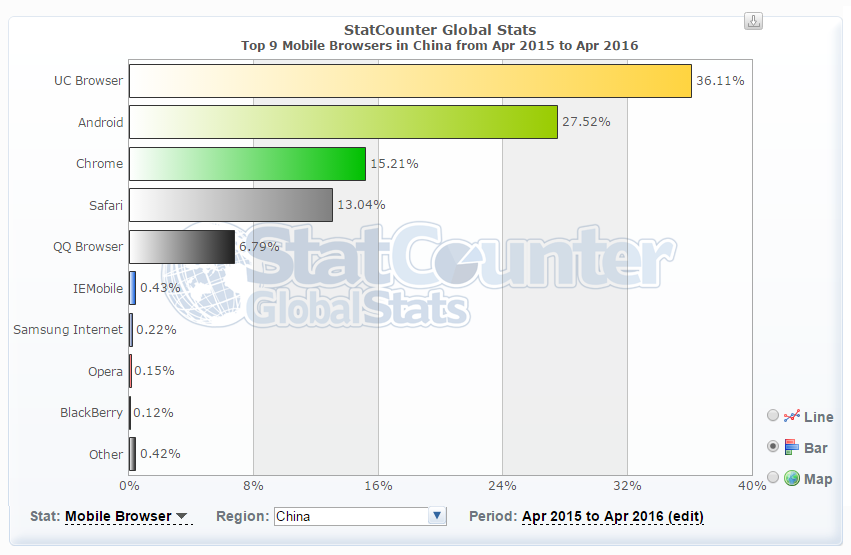
<http://gs.statcounter.com/#mobile_browser-CN-monthly-201504-201604>

中国用户的浏览器市场份额：

UC、Android内置、Chrome、Safari、QQ Browser都是webkit内核，从图上看占了98%的市场份额。



所以一定要伺候好-webkit-。 有的公司干脆只兼容-webkit-，别的兼容比如-ms-都不写。



测试每个CSS、JS的兼容性：

[caniuse.com](http://caniuse.com/)



CSS3率先在手机端找到生长的土地了，手机端的页面大胆放心的使用CSS3属性、选择器。

大胆点，不要怕。嗷嗷用各种新技术。手机的更新率非常高，基本上2年换一次手机。

# 三、百分比布局

## 3.1 相对于谁

思维线：

手机屏幕小 → 不希望有留白 → 希望撑满页面 → 视口宽度不同 → 不能用px写死 → 用百分比写宽度

百分比布局也叫作流式布局、弹性盒布局。

**百分比能够设置的属性是width、height、padding、margin。**

其他属性比如border、font-size不能用百分比设置的。

百分比是一个相对单位，相对谁的单位？父元素。如果没有父元素，实际上也有父元素就是body，就是屏幕。

width ：60%; 父元素的width的百分之60

height: 60%; 父元素的height的百分之60

padding:60%; 父元素的width的百分之60 ， 无论padding是什么方向的，竖直方向也是父元素的width

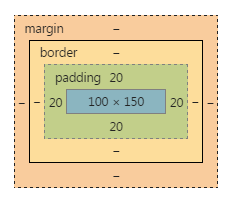
margin:60%; 父元素的width的百分之60 ， 无论margin是什么方向的，竖直方向也是父元素的width

小题目：

|  |
| --- |
| 1. div{ 2. width:200px; 3. height:300px; 4. padding:10px; 5. } 6. div p{ 7. width:50%; 8. height:50%; 9. padding:10%; 10. } |

此时p的真实宽度是140px\*190px

p的盒模型图：



## 3.2 CSS3弹性盒属性display:-webkit-box;

来看看百分比布局有什么不方便的地方，比如让三个p按1:2:3设置宽度。那应该给每个p设置多少的%。

|  |
| --- |
| 1. <div> 2. <p></p> 3. <p></p> 4. <p></p> 5. </div> |

1+2+3 = 6。 所以一份就是100/6 = 16.6666。

第一个p占了1份，就是16.6666%

第二个p占了2份，就是33.3333%

第三个p占了3份，就是50%

width不直观

|  |
| --- |
| 1. p:nth-child(1){ 2. float: left; 3. **width: 16.6666%;**   }   1. p:nth-child(2){ 2. float: left; 3. **width: 33.3333%;** 4. } 5. p:nth-child(3){ 6. float: left; 7. **height: 100px;** 8. } |

css3推出了一个新的属性，display属性有了一个新的值：box。目前还需要加兼容前缀display:-webkit-box;

加上这个属性之后，它自己的儿子都是弹性盒了。儿子们不需要设置float:left就能够默认并排。通过设置-webkit-box-flex:数字；来设置占的比例。

|  |
| --- |
| 1. div{ 2. /\*自己的亲儿子都是弹性盒模型了\*/ 3. /\*这个div是弹性盒模型的容器\*/ 4. **display: -webkit-box;** 5. } 6. p:nth-child(1){ 7. /\*box-flex属性表示比例\*/ 8. **-webkit-box-flex:1;** 9. height: 100px; 10. background: orange; 11. } 12. p:nth-child(2){ 13. /\*box-flex属性表示比例\*/ 14. **-webkit-box-flex:2;** 15. height: 100px; 16. background: skyblue; 17. } 18. p:nth-child(3){ 19. /\*box-flex属性表示比例\*/ 20. **-webkit-box-flex:3;** 21. height: 100px; 22. background: yellowgreen; 23. } |

总结一下：

|  |
| --- |
| 1. <div> **→ display:-webkit-box;** 2. <p></p> **→ -webkit-box-flex:1;** 3. <p></p>  **→ -webkit-box-flex:2;** 4. <p></p> **→ -webkit-box-flex:1;** 5. </div> |

## 3.3 CSS3新的盒模型box-sizing:border-box

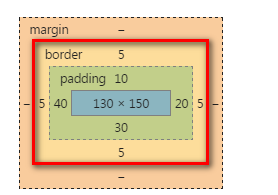
|  |
| --- |
| 1. div{ 2. width: 200px; 3. height: 200px; 4. padding: 10px 20px 30px 40px; 5. border:5px solid red; 6. **box-sizing:border-box;** 7. } |

box-sizing就是盒子的尺寸，这个属性定义了你设置的width、height指的是什么的宽度、高度。

默认来说，width、height是内容的宽度、高度。但是加上

|  |
| --- |
| 1. box-sizing:border-box; |

设置了我们的width、height指的是border的外侧构成的盒子的宽度和高度。



box-sizing:border-box；在流式布局中太好用了，因为border必须用px写单位，而加边框必须减宽度。而百分比的宽度，怎么减px的单位呢？？box-sizing:border-box;就可以让这个元素不用减border，因为width里面已经含有了border。

CSS3也新推出了一个样式表计算器，所有的属性可以用calc“函数”，这里的数值甚至可以百分比键数值：

|  |
| --- |
| 1. width: **calc(**50% - 10px**)**; |

calc只能加、减， 要带着单位运算。

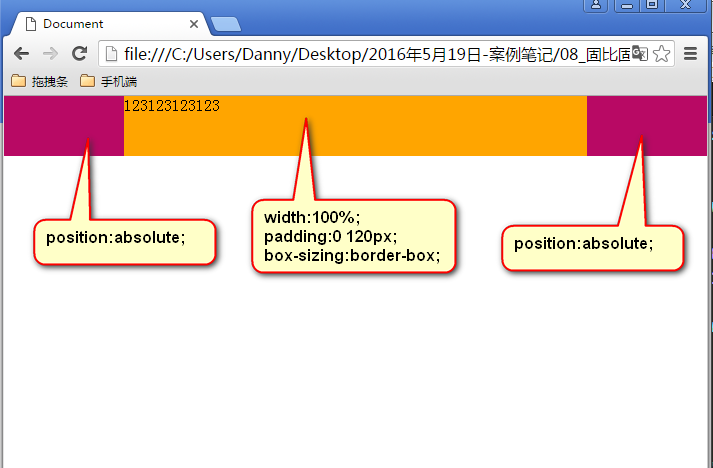
calc现在基本上没有人用了，都是box-sizing:border-box; 一招鲜吃遍天。

## 3.4 固比固

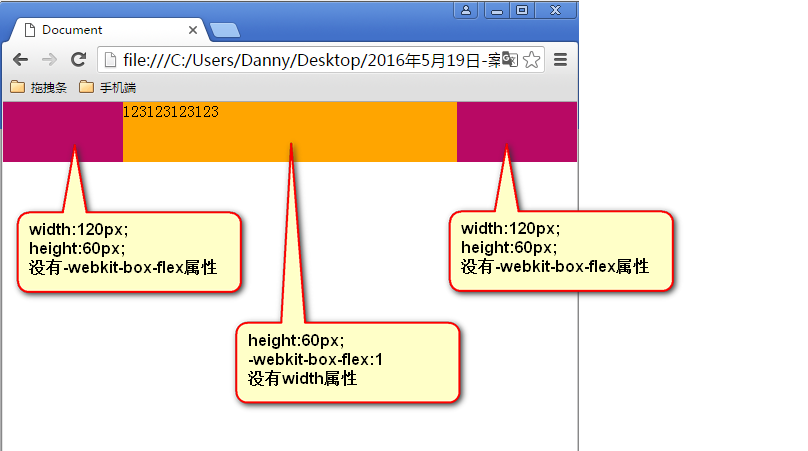
页面上经常有这样的部分：

|  |  |  |
| --- | --- | --- |
| 固定宽度 | 剩余部分 | 固定宽度 |

方法1：绝对定位的方法



方法2： display:box;属性



## 3.5 min-width、max-width

|  |
| --- |
| 1. header{ 2. width: 60%; 3. height: 100px; 4. background-color: red; 5. min-width: 360px; 6. max-width: 600px; 7. margin: 0 auto; 8. } |

比例的一个限制，最大是600px，最小是360px宽度。

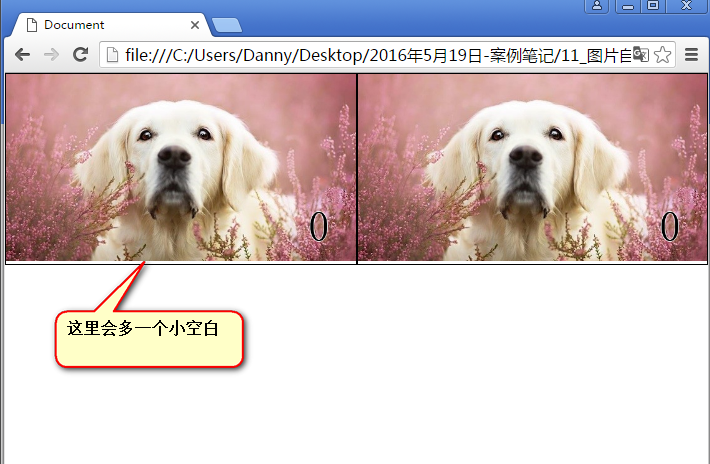
# 四、高度等比变化

做手机页面的时候，盒子可能是宽度是百分比写的，高度是固定的。有些时候，盒子的高度也会等比例变化。

## 4.1 图片自动就是等比例变化的

图片只设置width。height不设置，高度就是自动等比例变化的。

<div><img /></div> div是浮动的，没有高度，宽度50%，高度会自动收缩。img设置宽度100%。



## 4.2 padding-top猫腻

padding-top也是根据父亲的width当参考的。

|  |
| --- |
| 1. <style type="text/css"> 2. div{ 3. width: 50%; 4. **height: 0px;** 5. **padding-top:26.79%;** 6. border: 1px solid #000; 7. background: url(images/0.jpg); 8. background-size: contain; 9. } 10. </style> |

height的参考者是父盒子的height属性，而padding-top参考的是父盒子的width属性。

padding-top和width都有相同的参考。

原图是560px \* 300px ， width设置为50%的时候，就有等式：



解得x=26.79%;

## 4.3 js

利用js得到当前的计算后的宽度，然后设置height值，按比例设置。

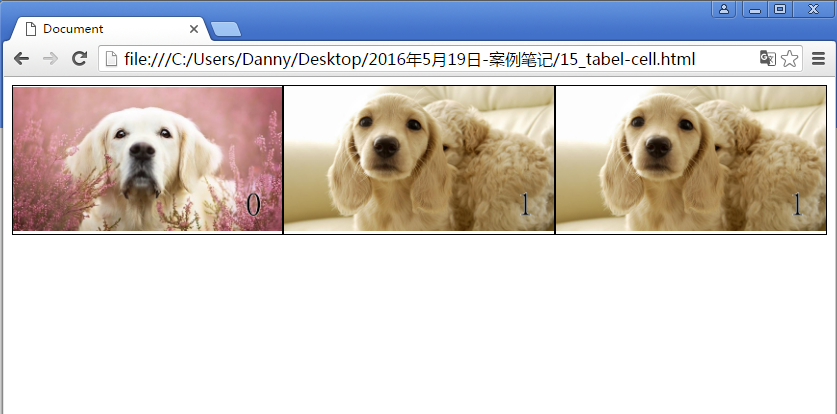
|  |
| --- |
| 1. //通过选择器得到元素们 2. var boxs = document.querySelectorAll(".box"); 3. //屏幕尺寸被改变的时候执行 4. window.onresize = set; 5. set(); 6. function set(){ 7. //var windowWidth = document.documentElement.clientWidth; 8. //盒子的计算后的宽度，找到0号盒子当代表，看看此时计算后的宽度是什么 9. **var boxWidth = parseInt(getComputedStyle(boxs[0])["width"]);** 10. //设置高度 11. **boxs[0].style.height = boxWidth / (560 / 300) + "px";** 12. **boxs[1].style.height = boxWidth / (560 / 300) + "px";** 13. } |

## 4.4 display:table-cell

display属性在css3中又新增了一种，叫做table-cell属性。设置这个属性的盒子就具备了像表格td一样的性质：

当里面有img的时候会自动的按相同的比例设置每个div的宽度。

|  |
| --- |
| 1. section{ 2. width: 100%; 3. **display: table;** 4. } 5. section div{ 6. **display: table-cell;** 7. border: 1px solid #000; 8. } 9. section div img{ 10. width: 100%; 11. } |

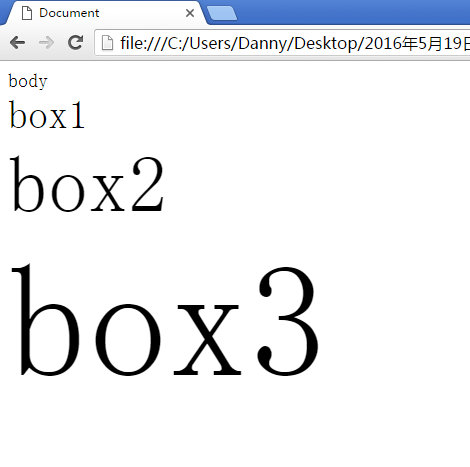


display:table-cell; 的元素可以接收vetical-align属性，值可以是top、middle、bottom。

|  |
| --- |
| 1. display: table-cell; 2. vertical-align: middle; |

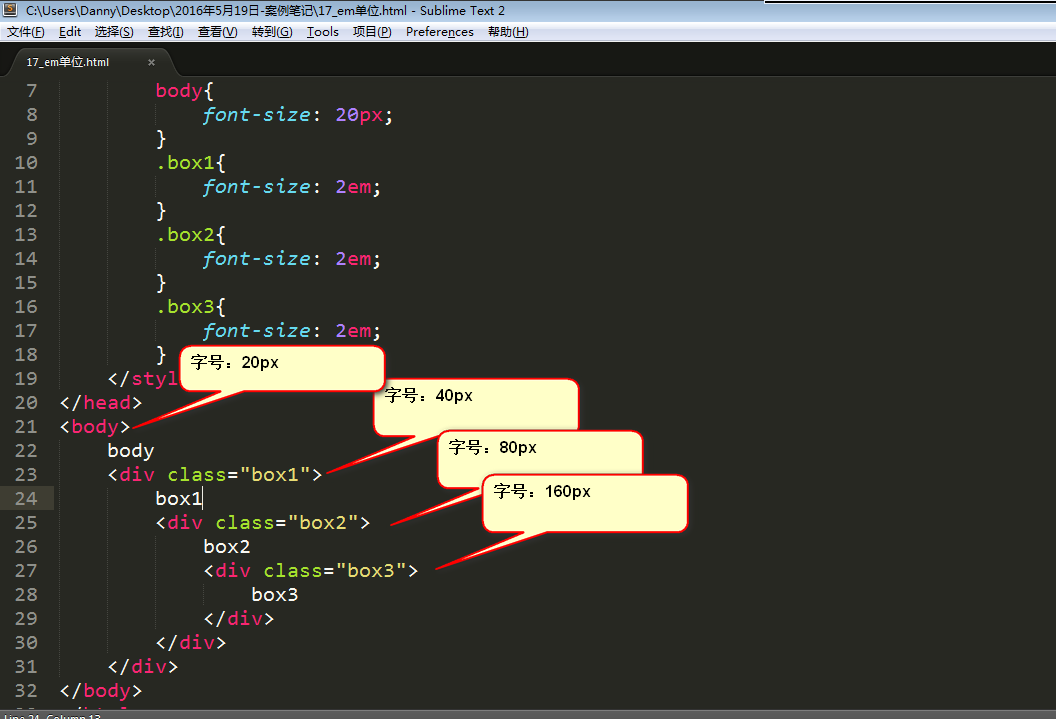
# 五、em和rem

|  |
| --- |
| 1. body 2. <div class="box1"> → font-size:2em; 3. box1 4. <div class="box2"> → font-size:2em; 5. box2 6. <div class="box3"> → font-size:2em; 7. box3 8. </div> 9. </div> 10. </div> |



**em表示父元素的字号的倍数。(特例：在text-indent属性中，表示文字宽度。)**

em为单位的时候，font-size属性是计算后继承，box1计算出来是40px。那么里面的box2、box3继承的都是40px。



em单位不仅仅可以用来设置字号，还可以设置任何盒模型的属性，比如width、height、padding、margin、border

|  |
| --- |
| 1. width: 30em; 2. height: 10em; |

1个em是多少？要看父盒子的字号。父盒子字号是16px，那么此时这个盒子的真实宽度高度： 480px\*160px;

em属性IE6也兼容，没有被开发出来，因为这个属性不好用，是以父亲的字号为基准。

|  |
| --- |
| 1. <!DOCTYPE html> 2. <html lang="en"> 3. <head> 4. <meta charset="UTF-8"> 5. <title>Document</title> 6. <style type="text/css"> 7. body{ 8. font-size: 16px; 9. } 11. .box1{ 12. background-color: pink; 13. width: 30em; 14. height: 10em; 15. font-size: 20px; 16. } 17. .box2{ 18. background-color: gold; 19. width: 3em; 20. height: 3em; 21. } 22. </style> 23. </head> 24. <body> → 字号是16px 25. <div class="box1"> → 宽度30em就是480px，高度10em就是160px。设置了字号是20px。 26. <div class="box2"></div> → 宽度3em，基准是父亲的字号20px，所以宽度60px 27. </div> 28. </body> 29. </html> |

**CSS3中，IE9开始兼容一个单位叫做rem。r就是root。**

这个单位也是倍数，是html标签字号的倍数。不看父亲，一律看html标签。

**html标签中，font-size设置为16px。那么全页面都是以16px当做基准，也就是说1rem=16px。**

em和rem区别非常大。em看父亲的字号，rem看html标签的字号。

设计师给你一个手机网页的设计图，这个设计图一定是定宽的。

比如这个设计图是600px宽度，logo 120px ， nav 弹性的 ， btn 120px  
此时有经验的前端开发工程师，一定不会用px当做单位，会进行计算。

比如html标签的font-size:16px; 那么120/16 = 7.5 ,所以logo的width就是7.5rem。

|  |
| --- |
| 1. header .logo{ 2. width: 7.5rem; 3. height: 2.5rem;   } |

这么做的好处是，可以配合媒体查询，来进行显示效果的增强，让大屏幕用户看到的东西真实的变大：

|  |
| --- |
| 1. @media(min-width:320px){ 2. **html{** 3. **font-size: 14px;** 4. **}** 5. } 6. @media(min-width:380px){ 7. **html{** 8. **font-size: 16px;** 9. **}** 10. } 11. @media(min-width:440px){ 12. **html{** 13. **font-size: 18px;** 14. **}** 15. } |

# 六、媒体查询

**css现在可以根据屏幕的宽度来设置应用哪些css，不应用哪些css。**

|  |
| --- |
| 1. **@media screen and (min-width:1200px){** 2. body{ 3. background-color: pink; 4. } 5. **}** 6. **@media screen and (min-width:700px) and (max-width:1200px){** 7. body{ 8. background-color: blue; 9. } 10. **}** 11. **@media screen and (max-width:700px){** 12. body{ 13. background-color: orange; 14. } 15. **}** |

IE9开始兼容。

@media就表示媒体查询，查询现在看这个网页的设备是什么，以及它的宽度是多少。

screen表示看这个网页的设备是显示器，而不是残疾人听力设备、也不是打印机。

后面用and符号罗列所有的可能性。

看下面的例子：

|  |
| --- |
| 1. **@media screen and (min-width:700px) and (max-width:1200px){** 2. body{ 3. background-color: pink; 4. } 5. **}** |

min-width表示应用里面这些样式的最小宽度是700px。换成人话，就是说大于等于700px的时候设置背景为粉色。

max-width表示应用里面这些样式的最大宽度是1200px。换成人话，就是说小于等于1200px的时候设置背景为粉色。

综合结果就是700~1200设置为粉色。

媒体查询只能包裹选择器，不能包裹k:v对儿。

IE6、7、8不支持媒体查询，也为了防止手机端的某些浏览器不支持媒体查询，所以不要把所有的选择器都放在媒体查询里面。“留活口”。

外链样式表也可以在link标签上应用媒体查询：

|  |
| --- |
| 1. <link href=”1.css” **media=”(min-width:400px) and (max-width:700px)”** rel=”stylesheet” /> |

# 七、携程建设作业

手机的真实像素，和手机的视口像素基本上是2.几 : 1

比如iPhone6plus的真实物理像素是1080px，视口是414px。基本就是2.几比1。

也就是说，你写了一个div，它的width是200px，那么机器要用大约400多个真实物理像素当做width。

所以，一张图片width:100px; 而真实物理像素在一行有200多个，所以你的图片如果是200px宽度，然后用background-size:100px auto; 那么图片就更精确。