一：CSS3兼容：

1. 浏览器支持程度差，需要添加私有前缀(移动端)
   1. requestFullScreen
   2. webkitRequestFullScreen
   3. mozRequestFullScreen
   4. msRequestFullScreen
   5. oRequestFullScreen

2、移动端支持优于PC端

3、不断改进中

4、应用相对广泛

二：选择器：

CSS3新增了许多灵活查找元素的方法，极大的提高了查找元素的效率和精准度。CSS3选择器与jQuery中所提供的绝大部分选择器兼容

1. 属性选择器：
2. E[attribute] 表示存在attr属性即可；

div[class]

1. E[attr=val] 表示属性值完全等于val；

div[class=mydemo]

1. E[attr\*=val] 表示的属性值里包含val字符并且在“任意”位置；

div[class\*=mydemo]

1. E[attr^=val] 表示的属性值里包含val字符并且在“开始”位置；

div[class^=mydemo]

1. E[attr$=val] 表示的属性值里包含val字符并且在“结束”位置；

div[class$=demos]

1. 伪类选择器-伪元素选择器：
   1. 之前学习的：a:hover a:link a:active a:visited
   2. 以某元素相对于其父元素或兄弟元素的位置来获取无素的结构伪类
2. E:first-child:查找E这个元素的父元素的第一个子元素E
3. E:last-child:最后一个子元素
4. E:nth-child(n): 第n个子元素，计算方法是E元素的全部兄弟元素
5. E:nth-last-child(n): 同E:nth-child(n) 相似，只是倒着计算
6. E:nth-child(even): 所有的偶数
7. E:nth-child(odd): 所有的奇数
8. E:nth-of-type(n):指定类型
9. E:empty 选中没有任何子节点的E元素，注意，空格也算子元素
10. E:target 结合锚点进行使用，处于当前锚点的元素会被选中
11. 重点说明：n遵循线性变化，其取值0、1、2、3、4、... 但是当n<=0时，选取无效
12. 案例代码：

/\*第一个li元素\*/  
li:first-child{  
 color: red;  
}  
/\*最后一个元素\*/  
li:last-child{  
 color: green;  
}  
/\*获取第10个元素\*/  
li:nth-child(10){  
 color: orange;  
}  
/\*获取倒数第3个li元素\*/  
li:nth-last-child(3){  
 color: purple;  
}  
/\*获取索引顺序为6的倍数的li元素\*/  
li:nth-child(6n){  
 text-decoration: underline;  
 border: 1px solid red;  
}  
/\*获取所有索引为偶数的li元素\*/  
li:nth-child(even){  
 border: 1px solid black;  
}  
/\*获取前5个li元素\*/  
li:nth-child(-n+5){  
 background-color: #ddd;  
}

* 1. n可是多种形式：nth-child(2n)、nth-child(2n+1)、nth-child(-n+5)等

1. 伪元素选择器：
   1. 重点：E::before、E::after
      1. 是一个行内元素，需要转换成块:display:block float:\*\* position:
      2. 必须添加content,哪怕不设置内容，也需要content:””
      3. E:after、E:before 在旧版本里是伪类，在新版本里是伪元素，新版本下E:after、E:before会被自动识别为E::after、E::before，按伪元素来对待，这样做的目的是用来做兼容处理
      4. E::before: 定义在一个元素的内容之前插入[content](http://www.dreamdu.com/css/property_content/)属性定义的内容与样式
      5. E::after: 定义在一个元素的内容之后插入[content](http://www.dreamdu.com/css/property_content/)属性定义的内容与样式
      6. 注意：
         1. IE6、IE7与IE8（怪异模式Quirks mode）不支持此伪元素
         2. CSS2中 E:before或者E:after，是属于伪类的，并且没有伪元素的概念，CSS3中 提出伪元素的概念 E::before和E::after，并且归属到了伪元素当中，伪类里就不再存在E:before或者 E:after伪类
   2. E::first-letter文本的第一个字母或字(不是词组)
   3. E::first-line 文本第一行
   4. E::selection 可改变选中文本的样式

三：颜色设置： 11111111 00000000 0-255

HTML5中添加了一些新的颜色的表示方式

1.RGBA：说得简单一点就是在RGB的基础上加进了一个通道Alpha。RGBA在RGB的基础上多了控制alpha透明度的参数。以上R、G、B三个参数，正整数值的取值范围为：0 - 255。百分数值的取值范围为：0.0% - 100.0%。超出范围的数值将被截至其最接近的取值极限。并非所有浏览器都支持使用百分数值。A参数，取值在0~1之间，不可为负值。RGBA比元素设置CSS的透明度更好，因为单独的颜色可以在不影响整个元素的透明度，他不会影响到元素其他的属性，比如说边框，字体同时也不会影响到其他元素的相关透明度

1. **语法:**
   * R：红色值。正整数 | 百分数
   * G：绿色值。正整数 | 百分数
   * B：蓝色值。正整数| 百分数
   * A：透明度。取值0~1之间
2. **使用示例：**

div{  
 width: 200px;  
 height: 200px;  
 background-color: rgba(10,20,245,0.5);  
 color: white;  
}

2. HSLA(H,S,L,A)：

1. **此色彩模式与[HSL](http://www.css88.com/book/css/values/color/hsl.htm)相同，只是在[HSL](http://www.css88.com/book/css/values/color/hsl.htm)模式上新增了Alpha透明度**
2. **语法：**
   * H：Hue(色调,色相)。0(或360)表示红色，120表示绿色，240表示蓝色，也可取其他数值来指定颜色。取值为：0 – 360,过渡为：(红橙黄绿青蓝紫红)
   * S：Saturation(饱和度)。取值为：0.0% - 100.0%
   * L：Lightness(亮度)。取值为：0.0% - 100.0%，50%是平衡值
   * A：Alpha透明度。取值0~1之间。
3. **示例**

span{  
 width: 200px;  
 height: 200px;  
 display: block;  
 background-color: hsla(360,100%,50%,0.6);  
 color: white;  
}

**3.关于透明度的补充说明：**

1. opacity只能针对整个盒子设置透明度，子盒子及内容会继承父盒子的透明度
2. transparent 不可调节透明度，始终完全透明
3. 使用rgba 来控制颜色，相对opacity ，不具有继承性

四：文本(shadow阴影)：

[text-shadow](http://www.w3.org/Style/Examples/007/text-shadow.en.html" \l "translations" \t "_blank)还没有出现时，大家在网页设计中阴影一般都是用photoshop做成图片，现在有了css3可以直接使用[text-shadow](http://www.w3.org/Style/Examples/007/text-shadow.en.html" \l "translations" \t "_blank)属性来指定阴影。这个属性可以有两个作用，产生阴影和模糊主体。这样在不使用图片时能给文字增加质感

1. **语法：**

**text-shadow ： none | <length> none | [<shadow>, ] \* <shadow> 或none | <color> [, <color> ]\***

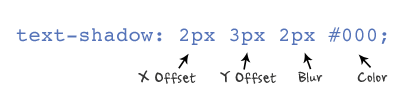
**也就是：**

**text-shadow:[颜色(Color) x轴(X Offset) y轴(Y Offset) 模糊半径(Blur)],[颜色(color) x轴(X Offset) y轴(Y Offset) 模糊半径(Blur)]...**

**或者**

**text-shadow:[x轴(X Offset) y轴(Y Offset) 模糊半径(Blur) 颜色(Color)],[x轴(X Offset) y轴(Y Offset) 模糊半径(Blur) 颜色(Color)]...**

1. **取值：**
   1. <length>：长度值，可以是负值。用来指定阴影的延伸距离。其中X Offset是水平偏移值，Y Offset是垂直偏移值
   2. <shadow>：阴影的模糊值，不可以是负值，用来指定模糊效果的作用距离
   3. <color>：指定阴影颜色，也可以是rgba透明色。
   4. 图示：



1. **说明：**

可以给一个对象应用一组或多组阴影效果，方式如前面的语法显示一样，用逗号隔开。text-shadow: X-Offset Y-Offset Blur Color中X-Offset表示阴影的水平偏移距离，其值为正值时阴影向右偏移，如果其值为负值时，阴影向左偏移；Y-Offset是指阴影的垂直偏移距离，如果其值是正值时，阴影向下偏移反之其值是负值时阴影向顶部偏移；Blur是指阴影的模糊程度，其值不能是负值，如果值越大，阴影越模糊，反之阴影越清晰，如果不需要阴影模糊可以将Blur值设置为0；Color是指阴影的颜色，其可以使用rgba色

1. **一些效果案例代码：**

.demo{  
 width: 600px;  
 padding: 30px;  
 background-color: #666;  
 margin:20px auto;  
 text-align: center;  
 font:bold 80px/100% "微软雅黑";  
 color: #fff;  
}  
/\*侧阴影效果\*/  
.demo1{  
 text-shadow: 2px 2px 2px #ff0000;  
}  
/\*辉光效果\*/  
.demo2{  
 text-shadow: 0 0 30px red;  
}  
/\*多层辉光效果\*/  
.demo3{  
 text-shadow:0 0 5px #fff,0 0 15px #fff,0 0 40px #fff ,0 0 70px red ;  
}  
/\*苹果经典效果\*/  
.demo4{  
 color: black;  
 text-shadow: 0 1px 1px #fff;  
}  
/\*浮雕效果\*/  
.demo5{  
 color: #ccc;  
 text-shadow: -1px -1px 0px #fff,-2px -2px 0px #eee,1px 1px 0px #444,2px 2px 0px #333;  
}  
/\*模糊字效果\*/  
.demo6{  
 color: transparent; /\*将本身设置为透明\*/  
 text-shadow: 0 0 6px #ff9966;  
}

五：边框圆角

border-radius可以通过值来定义样式相同的角，也对每个角分别定义

1.值的说明：

border-radius：\*px: 将创建四个大小一样的圆角

border-radius:\*px \*px \*px \*px: 四个值分别表示左上角、右上角、右下角、左下角

border-radius:\*px \*px：第一个值表示左上角、右下角；第二个值表示右上角、左下角

border-radius:\*px \*px \*px：第一个值表示左上角；第二个值表示右上角、左下角；第三个值表示右下角

2.单个圆角的设置：除了同时设置四个圆角以外，还可以单独对每个角进行设置。对应四个角，CSS3提供四个单独的属性：

\* border-top-left-radius

\* border-top-right-radius

\* border-bottom-right-radius

\* border-bottom-left-radius

这四个属性都可以同时设置1到2个值。如果设置1个值，表示水平半径与垂直半径相等。如果设置2个值，第一个值表示水平半径，第二个值表示垂直半径

3.补充：创建两个值的非对称圆角还有一种书写方式：如border-radius:20px/10px;表示在水平方向上20px,在垂直方向上10px;具体说明如下：可分别设置长、短半径，以“/”进行分隔，遵循“1，2，3，4”规则，“/”前面的1~4个用来设置横轴半径（分别对应横轴1、2、3、4位置 ），“/”后面1~4个参数用来设置纵轴半径（分别对应纵轴1、2、3、4位置 ）

4.示例代码：

div{  
 width: 200px;  
 /\*height: 200px;\*/  
 height: 100px; /\*为椭圆设置的高度\*/  
 background-color: red;  
 margin: 100px auto;  
 /\*四个方向的圆角值相同\*/  
 /\*border-radius: 100px;\*/  
 /\*左上，右下为20px 右上，左下为40px\*/  
 /\*border-radius: 20px 40px;\*/  
 /\*左上为20px 右上，左下为40px 右下为80px\*/  
 /\*border-radius: 20px 40px 80px;\*/  
 /\*左上：20px 右上：40px 右下：80px 左下：100px\*/  
 /\*border-radius: 20px 40px 80px 100px;\*/  
  
 /\*设置某一个方向上的圆角\*/  
 /\*border-bottom-left-radius: 100px;\*/  
  
 /\*非对称圆角--制作椭圆\*/  
 border-top-left-radius: 100px 50px;  
 border-top-right-radius: 100px 50px;  
 border-bottom-right-radius: 100px 50px;  
 border-bottom-left-radius: 100px 50px;  
  
 /\*非对称圆角的简写方式\*/  
 /\*四个方向上都是一致的非对称圆角\*/  
 /\*border-radius: 100px/50px;\*/  
 /\*也可以这样\*/  
 /\*border-radius: 100px 80px 60px 20px/20px 60px 80px 100px;\*/  
 /\*等价于：\*/  
 /\*border-top-left-radius: 100px 20px;  
 border-top-right-radius: 80px 60px;  
 border-bottom-right-radius: 60px 80px;  
 border-bottom-left-radius: 20px 100px;\*/  
}

九：边框阴影：

* + 1. box-shadow 属性向边框添加一个或多个阴影。
    2. 语法：box-shadow: h-shadow v-shadow blur spread color inset;

注释：box-shadow 向框添加一个或多个阴影。该属性是由逗号分隔的阴影列表，每个阴影由 2-4 个长度值、可选的颜色值以及可选的 inset 关键词来规定。省略长度的值是 0。

* + 1. 属性值说明：

|  |  |
| --- | --- |
| **值** | **描述** |
| *h-shadow* | 必需。水平阴影的位置。允许负值。 |
| *v-shadow* | 必需。垂直阴影的位置。允许负值。 |
| *blur* | 可选。模糊距离。 |
| *spread* | 可选。阴影的尺寸。值越大，阴影的扩散面积越大 |
| *color* | 可选。阴影的颜色。 |
| inset | 可选。将外部阴影 (outset) 改为内部阴影。 |

* + 1. 在添加阴影的时候，最好考虑到不同浏览器的兼容
    2. 案例示例：

/\*为item添加边框阴影\*/  
div[class=item]:nth-child(-n+4){  
 box-shadow: 1px 1px 3px 0px #bbb;  
}  
/\*可以同时添加多个边框阴影\*/  
div[class=item]:last-child{  
 box-shadow: 1px 1px 2px #ff0000 inset,-1px -1px 2px #ff0000 inset;  
}

十：边框图片

1. 功能：将图片规定为包围 div 元素的边框
   1. 定义和用法： border-image 属性是一个简写属性，用于设置以下属性

border-image-source

border-image-slice

border-image-width

border-image-outset

border-image-repeat

* 1. 属性说明：

|  |  |
| --- | --- |
| 值 | 描述 |
| border-image-source | 用在边框的图片的路径。 |
| border-image-slice | 图片边框向内偏移—裁切。 |
| border-image-width | 图片边框的宽度。 |
| border-image-outset | 边框图像区域超出边框的量。 |
| border-image-repeat | 图像边框是否应平铺(repeated)、铺满(rounded)或拉伸(stretched)。 |

1. 案例：任意拉伸的按钮：通过一个按钮的背景图片制作出任意大小的按钮，实现背景填充效果：
   * + 1. 效果：

* + - 1. 代码：

.downLoad{  
 width:80px;  
 height: 30px;  
 display: block;  
 text-align: center;  
 line-height: 30px;  
 border: 1px solid #ccc;  
 /\*设置图片边框背景\*/  
 border-image: url("../images/btn\_bg.png");  
 /\*设置裁切区域，同时设置填充模式\*/  
 border-image-slice: 10 fill;  
 /\*设置边框的大小，这个一般与裁切区域大小一致，否则就发生缩放\*/  
 border-image-width: 10px;  
 /\*设置边框的重复模式\*/  
 border-image-repeat: round;  
}

第四天

一：渐变：

渐变是CSS3当中比较丰富多彩的一个特性，通过渐变我们可以实现许多炫丽的效果，有效的减少图片的使用数量，并且具有很强的适应性和可扩展性。可分为线性渐变、径向渐变

1. linear-gradient线性渐变指沿着某条直线朝一个方向产生渐变效果
   1. 语法：

linear-gradient( [<point> || <angle>,]? <stop>, <stop> [, <stop>]\* )

* 1. 参数说明：
     + 第一个参数表示线性渐变的方向，
       1. to left：设置渐变为从右到左。相当于: 270deg;
       2. to right：设置渐变从左到右。相当于: 90deg;
       3. to top：设置渐变从下到上。相当于: 0deg;
       4. to bottom：设置渐变从上到下。相当于: 180deg。这是默认值，等同于留空不写。也可以直接指定度数，如45deg
     + 第二个参数是起点颜色,可以指定颜色的位置
     + 第三个参数是终点颜色，你还可以在后面添加更多的参数，表示多种颜色的渐变
  2. 示例：

div{  
 width: 400px;  
 height: 400px;  
 margin: 100px auto;  
 background: linear-gradient(0deg,red,orange,yellow,green, #00ffff,blue,purple);  
}

1. radial-gradient径向渐变指从一个中心点开始沿着四周产生渐变效果
   1. 语法：

<radial-gradient> = radial-gradient([ [ <shape> || <size> ] [ at <position> ]? , | at <position>, ]?<color-stop>[ , <color-stop> ]+)

* 1. 取值：
     1. <position> 确定圆心的位置。如果提供2个参数，第一个表示横坐标，第二个表示纵坐标；如果只提供一个，第二值默认为50%，即center
     2. shape：渐变的形状，ellipse表示椭圆形，circle表示圆形。默认为ellipse，如果元素形状为正方形的元素，则ellipse和circle显示一样
     3. size：渐变的大小，即渐变到哪里停止，它有四个值。 closest-side：最近边； farthest-side：最远边； closest-corner：最近角； farthest-corner：最远角。默认是最远的角farthest-corner
     4. <color>：指定颜色。Rgba hsla
  2. 一些案例示例代码：

.div1{  
 width: 200px;  
 height: 200px;  
 margin: 10px auto;  
 /\*设置径向渐变效果:从中心点开始，从一种颜色到另外一种颜色\*/  
 background: radial-gradient(circle at center,red,blue);  
 }  
 .div2{  
 width: 200px;  
 height: 200px;  
 border-radius: 100px;  
 margin: 10px auto;  
 /\*设置径向渐变效果:从指定坐开始，从一种颜色到另外一种颜色\*/  
 background: radial-gradient(circle at 50px 50px,#eeffff,#334455);  
 }  
 .div3{  
 width: 200px;  
 height: 200px;  
 border-radius: 100px;  
 margin: 10px auto;  
 /\*设置径向渐变效果:从指定坐标开始，从一种颜色到另外一种颜色，同时指定颜色的位置\*/  
 background: radial-gradient(circle at 50px 50px,#eeffff 0%,#666 70%,rgba(33,33,33,0.8) 80%);  
 }  
 /\*指定渐变的形状\*/  
 .div4{  
 width: 200px;  
 height: 100px;  
 margin: 10px auto;  
 /\*设置径向渐变效果:从中心点开始，从一种颜色到另外一种颜色\*/  
 background: radial-gradient(ellipse at center,red,green,blue);  
 }  
 /\*指定渐变的size\*/  
 .div5{  
 width: 200px;  
 height: 100px;  
 margin: 10px auto;  
 /\*设置径向渐变效果:从中心点开始，从一种颜色到另外一种颜色,同时指定了大小为渐变到最近的边\*/  
 background: radial-gradient(circle closest-side at center,red,green,blue);  
 }  
 /\*使用系统提供的位置设置\*/  
 .div6{  
 width: 200px;  
 height: 100px;  
 margin: 10px auto;  
 /\*设置径向渐变效果:从右上角点开始，从一种颜色到另外一种颜色\*/  
 background: radial-gradient(circle at top right,red,green,blue);  
 }  
</style>

二：背景：

A：background-size属性：

CSS里的background-size属性能够让程序员决定如何在指定的元素里展示,它通过改变背景尺寸的通过各种不同是属性值改变背景尺寸呈现的大小。往往建议不要将图放大，如果有需要，尽量让图缩小

1. 语法：

background-size: auto(原始图片大小) || number(数值) || percentage(百分比) || cover(放大铺满) || contain(缩小铺满)

1. 参数说明：
   1. auto：此值为默认值，保持背景图片的原始高度和宽度;
   2. number：此值设置具体的值，可以改变背景图片的大小;
   3. percentage：此值为百分值，可以是0%〜100%之间任何值，但此值只能应用在块元素上，所设置百分值将使用背景图片大小根据所在元素的宽度的百分比来计算。
   4. cover：此值是将图片放大，以适合铺满整个容器，这个主要运用在，当图片小于容器时，又无法使用background-repeat来实现时，我们就可以采用cover;将背景图片放大到适合容器的大小
   5. contain：此值刚好与cover相反，其主要是将背景图片缩小，以适合铺满整个容器，这个主要运用在，当背景图片大于元素容器时，而又需要将背景图片全部显示出来，此时我们就可以使用contain将图片缩小到适合容器大小为止。
   6. 当background-size取值为number和percentage时可以设置两个值，也可以设置一个值，当只取一个值时，第二个值相当于auto，但这里的auto并不会使背景图片的高度保持自己原始高度，而是会参照第一个参数值进行等比例缩放。
2. 使用案例：

<style>  
 \*{  
 padding: 0;  
 margin: 0;  
 }  
 .div1{  
 width: 200px;  
 height: 200px;  
 margin:10px auto;  
 /\*默认的背景设置，会让背景图片从左上角原点位置进行设置，不会自动的让背景图片适应容器的大小从而进行缩放\*/  
 background: url("../images/1.jpg");  
 }  
 .div2{  
 width: 200px;  
 height: 200px;  
 margin:10px auto;  
 background: url("../images/1.jpg");  
 /\*设置背景图片的大小,指定大小，有可能会让背景图片变形\*/  
 background-size: 200px 200px;  
 }  
 .div3{  
 width: 200px;  
 height: 200px;  
 margin:10px auto;  
 background: url("../images/1.jpg");  
 /\*设置背景图片的大小,cover:会让宽或者高适应当前容器的宽或者高，进行等比例缩放，但是超出的部分不会显示，所以有些图片的区域可能会无法显示\*/  
 background-size: cover;  
 }  
 .div4{  
 width: 200px;  
 height: 200px;  
 margin:10px auto;  
 background: url("../images/1.jpg") no-repeat;  
 /\*设置背景图片的大小,cover:会让宽或者高适应当前容器的宽或者高，进行等比例缩放，图片完全在容器以内，但是在不重复背景图片的情况下，会造成容器的部分区域空白\*/  
 background-size: contain;  
 }  
</style>

B: background-origin:

1. 作用：background-origin 属性规定 background-position 属性相对于什么位置来定位。默认值是left top左上角
2. 语法：background-origin: padding-box|border-box|content-box;
3. 属性值说明：

|  |  |
| --- | --- |
| padding-box | 背景图像相对于内边距框来定位。 |
| border-box | 背景图像相对于边框盒来定位。 |
| content-box | 背景图像相对于内容框来定位。 |

C: background-clip:

* + 1. background-clip 属性规定背景的绘制区：虽然是设置裁切，但是控制的是显示。说白了，就是设置最终显示那些区域
    2. 语法：background-clip: border-box|padding-box|content-box;
    3. 属性值说明：

|  |  |
| --- | --- |
| **值** | **描述** |
| border-box | 背景被裁剪到边框盒。 |
| padding-box | 背景被裁剪到内边距框。 |
| content-box | 背景被裁剪到内容框。 |

D:几个属性的综合案例：精灵图的使用。

1. 需求：为块设置精灵图背景，需要更大的展示区域，能够以更大的范围响应用户的需要，但是只需要显示指定的背景图片
2. 效果：

 > 

1. 代码：

.jd\_topEeum {  
 height: 44px;  
 width: 40px;  
 position: absolute;  
 background: url("../images/sprites.png") no-repeat;  
 background-clip: content-box;  
 background-origin: content-box;  
 background-size: 200px 200px;   
 padding: 12px;  
 top: 0;  
}

三：过渡：

通过过渡transition,我们可以在不使用 Flash 动画或 JavaScript 的情况下，当元素从一种样式变换为另一种样式时为元素添加效果. 要实现这一点，必须规定两项内容：1.规定希望把效果添加到哪个 CSS 属性上，2.规定效果的时长

1. 语法：

transition: property duration timing-function delay;

1. 参数说明：

transition 属性是一个简写属性，用于设置四个过渡属性：transition-property |transition-duration |transition-timing-function |transition-delay

|  |  |
| --- | --- |
| **值** | **描述** |
| [transition-property](http://www.w3school.com.cn/cssref/pr_transition-property.asp" \o "CSS3 transition-property 属性) | 规定设置过渡效果的 CSS 属性的名称。 |
| [transition-duration](http://www.w3school.com.cn/cssref/pr_transition-duration.asp" \o "CSS3 transition-duration 属性) | 规定完成过渡效果需要多少秒或毫秒。 |
| [transition-timing-function](http://www.w3school.com.cn/cssref/pr_transition-timing-function.asp" \o "CSS3 transition-timing-function 属性) | 规定速度效果的速度曲线。 |
| [transition-delay](http://www.w3school.com.cn/cssref/pr_transition-delay.asp" \o "CSS3 transition-delay 属性) | 定义过渡效果何时开始。 |

1. 补充说明tansition-timing-function: 属性规定过渡效果的速度曲线

|  |  |
| --- | --- |
| **值** | **描述** |
| linear | 规定以相同速度开始至结束的过渡效果（等于 cubic-bezier(0,0,1,1)）。 |
| ease | 规定慢速开始，然后变快，然后慢速结束的过渡效果（cubic-bezier(0.25,0.1,0.25,1)）。 |
| ease-in | 规定以慢速开始的过渡效果（等于 cubic-bezier(0.42,0,1,1)）。 |
| ease-out | 规定以慢速结束的过渡效果（等于 cubic-bezier(0,0,0.58,1)）。 |
| ease-in-out | 规定以慢速开始和结束的过渡效果（等于 cubic-bezier(0.42,0,0.58,1)）。 |
| cubic-bezier(*n*,*n*,*n*,*n*) | 在 cubic-bezier 函数中定义自己的值。可能的值是 0 至 1 之间的数值。 |

1. 案例说明

div{  
 width: 200px;  
 height: 200px;  
 background-color: red;  
 /\*添加单个过渡效果\*/  
 /\*transition:background-color 2s;\*/  
 /\*也可以同时设置多个过渡效果\*/  
 /\*transition:background-color 2s,left 1s;\*/  
 /\*可以设置某个过渡效果的延迟\*/  
 /\*transition:background-color 2s,left 1s 1s;\*/  
 /\*可以设置过渡效果的速率曲线\*/  
 /\*transition:background-color 2s,left 1s ease-out 1s;\*/  
 /\*还可以一次性的为所有属性添加过渡效果\*/  
 transition:all 1s;  
 position: absolute;  
 left: 0;  
 top: 0;  
}

1. 使用建议：

因为transition最早是有由webkit内核浏览器提出来的，mozilla和opera都是最近版本才支持这个属性，而我们的大众型浏览器IE全家都是不支持，另外由于各大现代浏览器Firefox,Safari,Chrome,Opera都还不支持W3C的标准写法，所以在应用transition时我们有必要加上各自的前缀，最好在放上我们W3C的标准写法，这样标准的会覆盖前面的写法，只要浏览器支持我们的transition属性，那么这种效果就会自动加上去，如

-moz-transition: all 5s ease 1s;  
-webkit-transition: all 1s ease 1s;  
-o-transition: all 1s ease 1s;  
transition: all 1s ease 1s;

第五天

一：2D转换：

通过 CSS3 transform转换，我们能够对元素进行移动、缩放、转动、拉长或拉伸.

1. 2D移动：translate()。使用translate()函数，你可以把元素从原来的位置移动。移动参照元素左上角原点
   1. 语法：translate(tx) | translate(tx,ty)
   2. tx是一个代表X轴（横坐标）移动的向量长度，当其值为正值时，元素向X轴右方向移动，反之其值为负值时，元素向X轴左方向移动。
   3. ty是一个代表Y轴（纵向标）移动的向量长度，当其值为正值时，元素向Y轴下方向移动，反之其值为负值时，元素向Y轴上方向移动。如果ty没有显式设置时，相当于ty=0。
   4. 也可以使用translateX(tx) 或者 translateY(ty)
   5. 案例示例：

div:hover{  
 /\*设置两个值，第一个参数表示X方向 第二个参数表示Y方向\*/  
 /\*transform: translate(100px,100px);\*/  
 /\*也可以只传入一个参数，表示X方向\*/  
 /\*transform: translate(100px);\*/  
 /\*也可以指定具体的方向,如下代码，表示Y方向正值方向上移动100px\*/  
 transform:translateY(100px);  
}

1. 2D缩放：scale()：缩放scale()函数让元素根据中心原点对对象进行缩放。默认的值1。因此0.01到0.99之间的任何值，使一个元素缩小；而任何大于或等于1.01的值，让元素显得更大.缩放是参照元素中心点。
   1. 语法：scale(sx|ty) | scale(sx,sy)
   2. sx：用来指定横向坐标（X轴）方向的缩放向量，如果值为0.01~0.99之间，会让对象在X轴方向缩小，如果值大于或等于1.01，对象在Y轴方向放大。
   3. sy：用来指定纵向坐标（Y轴）方向的缩放量，如果值为0.01~0.99之间，会让对象在Y轴方向缩小，如果值大于或等于1.01，对象在Y轴方向放大
   4. 也可以使用scaleX(sx) 或者scaleY(sy)
   5. 案例示例：

div:hover{  
 /\*传入两个值，第一个参数表示X方向的缩放 第二个参数表示Y方向上的缩放\*/  
 /\*transform: scale(2,0.5);\*/  
 /\*也可以只传入一个值，表示X方向和Y方向上相同的缩放\*/  
 /\*transform: scale(2);\*/  
 /\*也可以指定具体方向上的缩放\*/  
 transform: scaleX(2);  
}

1. 2D旋转：rotate()：旋转rotate()函数通过指定的角度参数对元素根据对象原点指定一个2D旋转。它主要在二维空间内进行操作，接受一个角度值，用来指定旋转的幅度。如果这个值为正值，元素相对原点中心顺时针旋转；如果这个值为负值，元素相对原点中心逆时针旋转
   1. 语法：rotate(a);
   2. a: 代表的是一个旋转的角度值。其取值可以是正的，也可以是负的。如果取值为正值时，元素默认之下相对元素中心点顺时针旋转；如果取值为负值时，元素默认之下相对元素中心点逆时针旋转
   3. 案例示例：

div:hover{  
 /\*传入旋转的角度，如果正值，则进行顺时针旋转\*/  
 /\*transform: rotate(90deg);\*/  
 /\*如果传入负值，则逆时针旋转\*/  
 transform: rotate(-270deg);  
}

1. 2D翻转：skew()：能够让元素倾斜显示。它可以将一个对象以其中心位置围绕着X轴和Y轴按照一定的角度倾斜。这与rotate()函数的旋转不同，rotate()函数只是旋转，而不会改变元素的形状。skew()函数不会旋转，而只会改变元素的形状
   1. 语法：skew(ax) | skew(ax,ay)
   2. ax：用来指定元素水平方向（X轴方向）倾斜的角度。
   3. ay：用来指定元素垂直方向（Y轴方向）倾斜的角度。如果未显式的设置这个值，其默认为０。
   4. 也可以使用skewX(sx) 或者 skewY(sy)
   5. 案例示例：

div:hover{  
 /\*在X方向上倾斜30度\*/  
 transform: skewX(30deg);  
}

1. transform-origin: 允许你改变被转换元素的位置。
   1. 示例：

div{  
 width: 100px;  
 height: 100px;  
 margin: 100px auto;  
 background-color: red;  
 /\*添加过渡\*/  
 transition:all .5s;  
 /\*设置缩放的中心，默认是元素中心位置,现修改为元素左上角\*/  
 transform-origin: 0px 0px;  
}  
div:hover{  
 transform: scale(2);  
}

二：3D转换：

三维变换使用基于二维变换的相同属性，可以让我们基于三个坐标方向对元素进行变换。和二维变形一样，三维变形可以使用transform属性来设置

1. 3D移动
   1. 方法：translate3d(x,y,z) 使元素在这三个纬度中移动,也可以分开写，如：translateX(length),translateY(length), translateZ(length)
   2. 示例：

div:hover{  
 /\*Y轴移动+100px\*/  
 /\*transform:translateY(100px);\*/  
 /\*X轴移动100px\*/  
 /\*transform:translateX(100px);\*/  
 /\*x轴和Y轴方向同时移动\*/  
 transform:translate3d(100px,100px,0px);  
}

1. 3D缩放
   1. scale3d(number,number,number) 使元素在这三个纬度中缩放，也可分开写，如：scaleX(),scaleY(),scaleZ()
   2. 示例

div:hover{  
 /\*Y轴方向放大1倍\*/  
 /\*transform: scaleX(2);\*/  
 /\*X轴方向缩小0.5\*/  
 /\*transform: scaleX(0.5);\*/  
 /\*x轴和Y轴方向同时进行缩放\*/  
 transform: scale3d(2,0.5,1);  
}

1. 3D旋转
2. rotate3d(x,y,z,angle)：指定需要进行旋转的坐标轴
3. rotateX(angle) 是元素依照x轴旋转；
4. rotateY(angle) 是元素依照y轴旋转；
5. rotateZ(angle) 是元素依照z轴旋转
6. 示例：

div:hover{  
 /\*Y轴方向旋转45度\*/  
 /\*transform: rotateY(45deg);\*/  
 /\*X轴方向旋转90度\*/  
 /\*transform: rotateX(90deg);\*/  
 /\*x轴和Y轴方向同时进行旋转放\*/  
 transform: rotate3d(1,1,0,45deg);  
}

1. 透视/景深效果

左手法则：大拇指指向当前坐标轴的下方向，手指环绕的方向就是正方向

1. perspective(length) 为一个元素设置三维透视的距离。仅作用于元素的后代，而不是其元素本身。当perspective:none/0;时，相当于没有设perspective(length)。比如你要建立一个小立方体，长宽高都是200px。如果你的perspective < 200px ，那就相当于站在盒子里面看的结果，如果perspective 非常大那就是站在非常远的地方看（立方体已经成了小正方形了），意味着perspective 属性指定了观察者与z=0平面的距离，使具有三维位置变换的元素产生透视效果
2. perspective-origin属性规定了镜头在平面上的位置。默认是放在元素的中心
3. transform-style：使被转换的子元素保留其 3D 转换(需要设置在父元素中)：

|  |  |
| --- | --- |
| **值** | **描述** |
| flat | 子元素将不保留其 3D 位置-平面方式。 |
| preserve-3d | 子元素将保留其 3D 位置—立体方式。 |

1. 案例：立方体

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
 <style>  
 \*{  
 padding: 0;  
 margin: 0;  
 }  
 .box{  
 width: 200px;  
 height: 200px;  
 margin:100px auto;  
 position: relative;  
 opacity: 0.5;  
 /\*设置一下默认的变换，否则由于观察角度在正面，无法看到立方体\*/  
 transform: rotateX(-30deg) rotateY(30deg);  
 /\*保持3D效果\*/  
 transform-style: preserve-3d;  
 }  
 /\*公共样式\*/  
 .front,.back,.left,.right,.top,.bottom{  
 width: 200px;  
 height: 200px;  
 position: absolute;  
 left: 0;  
 top: 0;  
 }  
 .left{  
 background: red;  
 transform: translateX(-100px) rotateY(-90deg);  
 }  
 .right{  
 background: green;  
 transform: translateX(100px) rotateY(90deg);  
 }  
 .front{  
 background: blue;  
 transform: translateZ(100px);  
 }  
 .back{  
 background: pink;  
 transform: translateZ(-100px);  
 }  
 .top{  
 background: orange;  
 transform: translateY(-100px) rotateX(90deg);  
 }  
 .bottom{  
 background: purple;  
 transform: translateY(100px) rotateX(-90deg);  
 }  
 </style>  
</head>  
<body>  
<div class="box">  
 <div class="front">front</div>  
 <div class="back">back</div>  
 <div class="left">left</div>  
 <div class="right">right</div>  
 <div class="top">top</div>  
 <div class="bottom">bottom</div>  
</div>  
</body>  
</html>

三：动画：

动画是CSS3中具有颠覆性的特征之一，可通过设置多个节点来精确控制一个或一组动画，常用来实现复杂的动画效果.

1. 必要元素：

a、通过@keyframes指定动画序列；自动补间动画，确定两个点，系统会自动计算中间过程。这两个点就称为关键帧。我们可以设置多个关键帧

b、通过百分比将动画序列分割成多个节点；

c、在各节点中分别定义各属性

d、通过animation将动画应用于相应元素；

1. animation样式常用属性：
2. 动画序列的名称:animation-name: move;
3. 动画的持续时间:animation-duration: 1s;
4. 动画的延时:animation-delay: 1s;
5. **播放状态:animation-play-state: paused|running;**
6. 播放速度:animation-timing-function: linear;
7. 播放次数 反复:animation-iteration-count: 1;
8. 动画播放完结后的状态:animation-fill-mode: forwards;
9. 循环播放时，交叉动画:animation-direction: alternate;
10. 代码说明：

<style>  
 \*{  
 padding: 0;  
 margin: 0;  
 }  
 div{  
 width: 300px;  
 height: 300px;  
 margin:100px auto;  
 }  
 div > img{  
 width:100%;  
 }  
 /\*添加动画\*/  
 @keyframes rotateAni {  
 0%{  
 /\*可以同时对多个属性添加动画效果\*/  
 transform: rotate(0deg) scale(1);  
 }  
 50%{  
 transform: rotate(180deg) scale(2);  
 }  
 100%{  
 transform: rotate(360deg) scale(1);  
 }  
 }  
 div:hover > img{  
 /\*动画名称-自定义\*/  
 animation-name: rotateAni;  
 /\*动画时间\*/  
 animation-duration: 1s;  
 /\*动画速率曲线： linear：匀速 ease：动画以低速开始，然后加快，在结束前变慢 ease-in：动画以低速开始 ease-out：动画以低速结束 ease-in-out：动画以低速开始和结束\*/  
 animation-timing-function: linear;  
 /\*动画播放次数\*/  
 animation-iteration-count: 4;  
 /\*动画时间延迟\*/  
 animation-delay: 0s;  
 /\*动画播放完的状态： forwards:保持动画播放完毕后的状态 backwards:退回到原始状态(默认值)\*/  
 animation-fill-mode: forwards;  
 /\*动画是否轮流反射播放： alternate:在规定的次数内轮流反射播放 normal:正常播放\*/  
 /\*animation-direction: alternate;\*/  
 }  
 div:active >img{  
 /\*动画的当前播放状态： paused：暂停 running:运行\*/  
 animation-play-state: paused;  
 }  
</style>

1. 动画案例：汽车/钟表/宇宙

四：列布局

CSS3中新出现的多列布局(multi-column)是传统HTML网页中块状布局模式的有力扩充。这种新语法能够让WEB开发人员轻松的让文本呈现多列显示。我们知道，当一行文字太长时，读者读起来就比较费劲，有可能读错行或读串行；人们的视点从文本的一端移到另一端、然后换到下一行的行首，如果眼球移动浮动过大，他们的注意力就会减退，容易读不下去。所以，为了最大效率的使用大屏幕显示器，页面设计中需要限制文本的宽度，让文本按多列呈现，就像报纸上的新闻排版一样

1. 常用属性：
   1. column-count: 属性设置列的具体个数
   2. column-width: 属性控制列的宽度
   3. column-gap: 两列之间的缝隙间隔
   4. column-rule: 规定列之间的宽度、样式和颜色
   5. column-span: 规定元素应横跨多少列(n:指定跨n列 all:跨所有列)
2. 多列的用法：

<style>  
 \*{  
 padding: 0;  
 margin: 0;  
 }  
 .wrapper{  
 width:1054px;  
 padding:20px;  
 margin:0 auto;  
 font-family: "微软雅黑",Arial;  
 /\*设置以几列的方式显示\*/  
 column-count:2;  
 /\*指定列宽\*/  
 column-width:500px;  
 /\*指定列与列之间的间距\*/  
 column-gap: 50px;  
 /\*指定列与列之间间隙的样式\*/  
 /\*column-rule:2px dotted red\*/  
 /\*相对应下面的三个属性\*/  
 column-rule-color:red;  
 column-rule-style:dotted;  
 column-rule-width:2px;  
 }  
 .wrapper > h4{  
 column-span: all;  
 }  
</style>

1. 列高度的平衡：

如果设定列的最大高度，这个时候，文本内容会从第一列开始填充，然后第二列，第三列

max-height: 300px;

五：伸缩布局

布局的传统解决方案，基于[盒状模型](https://developer.mozilla.org/en-US/docs/Web/CSS/box_model" \t "_blank)，依赖 [display](https://developer.mozilla.org/en-US/docs/Web/CSS/display" \t "_blank)属性 + [position](https://developer.mozilla.org/en-US/docs/Web/CSS/position" \t "_blank)属性 + [float](https://developer.mozilla.org/en-US/docs/Web/CSS/float" \t "_blank)属性。它对于那些特殊布局非常不方便。CSS3在布局方面做了非常大的改进，使得我们对块级元素的布局排列变得十分灵活，适应性非常强，其强大的伸缩性，在响应式开中可以发挥极大的作用。

1. 重要属性
   1. display:flex:如果一个容器设置了这个属性，那么这个盒子里面的所有直接子元素都会自动的变成伸缩项(flex item)
   2. **j**ustify-content: 设置或检索弹性盒子元素在主轴（横轴）方向上的对齐方式,语法：justify-content：flex-start | flex-end | center | space-between | space-around

* flex-start：弹性盒子元素将向行起始位置对齐。该行的第一个子元素的主起始位置的边界将与该行的主起始位置的边界对齐，同时所有后续的伸缩盒项目与其前一个项目对齐。
* flex-end：弹性盒子元素将向行结束位置对齐。该行的第一个子元素的主结束位置的边界将与该行的主结束位置的边界对齐，同时所有后续的伸缩盒项目与其前一个项目对齐。
* center：弹性盒子元素将向行中间位置对齐。该行的子元素将相互对齐并在行中居中对齐，同时第一个元素与行的主起始位置的边距等同与最后一个元素与行的主结束位置的边距（如果剩余空间是负数，则保持两端相等长度的溢出）。
* space-between：弹性盒子元素会平均地分布在行里。如果最左边的剩余空间是负数，或该行只有一个子元素，则该值等效于'flex-start'。在其它情况下，第一个元素的边界与行的主起始位置的边界对齐，同时最后一个元素的边界与行的主结束位置的边距对齐，而剩余的伸缩盒项目则平均分布，并确保两两之间的空白空间相等。
* space-around：弹性盒子元素会平均地分布在行里，两端保留子元素与子元素之间间距大小的一半。如果最左边的剩余空间是负数，或该行只有一个伸缩盒项目，则该值等效于'center'。在其它情况下，伸缩盒项目则平均分布，并确保两两之间的空白空间相等，同时第一个元素前的空间以及最后一个元素后的空间为其他空白空间的一半。
  1. flex-flow: flex-flow属性是flex-direction属性和flex-wrap属性的简写形式，默认值为row nowrap。它用来设置或检索弹性盒模型对象的子元素排列方式,它的两种取值：flex-direction：定义弹性盒子元素的排列方向。flex-wrap：控制flex容器是单行或者多行。
     + flex-direction： **flex-direction**：row | row-reverse | column | column-reverse
* row：主轴与行内轴方向作为默认的书写模式。即横向从左到右排列（左对齐）。
* row-reverse：对齐方式与row相反。
* column：主轴与块轴方向作为默认的书写模式。即纵向从上往下排列（顶对齐）。
* column-reverse：对齐方式与column相反。
  + - flex-wrap： flex-wrap：nowrap | wrap | wrap-reverse
      * nowrap：flex容器为单行。该情况下flex子项可能会溢出容器
      * wrap：flex容器为多行。该情况下flex子项溢出的部分会被放置到新行，子项内部会发生断行
      * wrap-reverse：反转 wrap 排列。

1. flex属性：flex属性是flex-grow, flex-shrink 和 flex-basis的简写，默认值为0 1 auto。后两个属性可选
   * + - 语法：flex: [flex-grow] [flex-shrink] [flex-basis]：大多数情况下没必要使用这种语法，当使用flex缩写时，大多数情况下没必要使用这种语法
       - flex: [number]：这个语法指定了一个数字，代表了这个伸缩项目该占用的剩余空间比例
       - flex: auto：属性值被设为auto的伸缩项目，会根据主轴自动伸缩以占用所有剩余空间
2. 案例-携程网页面：
   1. 主要技术点:display:flex; | flex-wrap:wrap;
   2. 效果：



* 1. div结构：

<div class="layout">  
 <header>  
 <div class="banner">  
 <a href="">  
 <img src="../images/banner.jpg" alt="">  
 </a>  
 </div>  
 </header>  
 <div class="main">  
 <div class="item">  
 <div class="pub"></div>  
 <div class="sub">  
 <a href="javascript:;">海外酒店</a>  
 <a href="javascript:;">团购</a>  
 <a href="javascript:;">特惠酒店</a>  
 <a href="javascript:;">客栈公寓</a>  
 </div>  
 </div>  
 <div class="item">  
 <div class="pub"></div>  
 <div class="sub">  
 <a href="javascript:;">海外酒店</a>  
 <a href="javascript:;">团购</a>  
 <a href="javascript:;">特惠酒店</a>  
 <a href="javascript:;">客栈公寓</a>  
 </div>  
 </div>  
 <section class="extra">  
 <a href="javascript:;">  
 <img src="../images/extra\_1.png">  
 </a>  
 <a href="javascript:;">  
 <img src="../images/extra\_2.png">  
 </a>  
 </section>  
 </div>  
 <footer>  
 <nav>  
 <a href="javascript:;">电话预订</a>  
 <a href="javascript:;">下载客户端</a>  
 <a href="javascript:;">我的</a>  
 </nav>  
 <!-- 版本信息 -->  
 <div class="copyright">  
 <!-- 网站链接 -->  
 <p class="link">  
 <a href="javascript:;">网站地图</a> |  
 <a href="javascript:;">ENGLISH</a> |  
 <a href="javascript:;">电脑版</a>  
 </p>  
 <p>&copy;2015 携程旅行</p>  
 </div>  
 </footer>  
</div>

* 1. 样式：

<style>  
 \*{  
 padding: 0;  
 margin: 0;  
 }  
 .layout{  
 min-width: 320px;  
 }  
 header{  
  
 }  
 .banner{  
 display: flex;  
 }  
 .banner > a{  
 flex: 1;  
 }  
 .banner > a > img{  
 width:100%;  
 height:auto;  
 display: block;  
 }  
  
 footer{  
 width:100%;  
 }  
 footer > nav{  
 display: flex;  
 }  
 footer > nav > a{  
 flex: 1;  
 text-align: center;  
 text-decoration: none;  
 color: #666;  
 font-size: 12px;  
 line-height:40px;  
 border-top:1px solid #ccc;  
 border-bottom:1px solid #ccc;  
 }  
 .copyright > p {  
 padding-top:10px;  
 text-align: center;  
 font-size: 12px;  
 }  
 .copyright > p > a{  
 padding:0 5px;  
 text-decoration: none;  
 color: #666;  
 }  
  
 .main{  
 padding:5px;  
 }  
 .main > .item{  
 margin-top: 5px;  
 display: flex;  
 height:90px;  
 border-radius: 5px;  
 }  
 .main > .item:nth-child(1){  
 background-color: #ff697a;  
 }  
 .main > .item > .pub{  
 flex: 1;  
 border-top:1px solid #fff;  
 }  
 .main > .item > .sub{  
 flex: 2;  
 display: flex;  
 flex-wrap: wrap;  
 }  
 .main > .item > .sub > a{  
 width:50%;  
 height: 45px;  
 text-align: center;  
 line-height:45px;  
 color: white;  
 text-decoration: none;  
 font-size: 13px;  
 border-left:1px solid #fff;  
 border-top:1px solid #fff;  
 box-sizing: border-box;  
 }  
 .main .extra{  
 display: flex;  
 }  
 .main .extra > a{  
 flex: 1;  
 }  
 .main .extra > a > img{  
 width:100%;  
 }  
</style>

六：web字体：

开发人员可以为自已的网页指定特殊的字体，无需考虑用户电脑上是否安装了此特殊字体，从此把特殊字体处理成图片的时代便成为了过去。它的支持程度比较好，甚至IE低版本浏览器也能支持。

1. 字体格式

不同浏览器所支持的字体格式是不一样的，我们有必要了解一下有关字体格式的知识。

1、TureTpe(.ttf)格式

.ttf字体是Windows和Mac的最常见的字体，是一种RAW格式，支持这种字体的浏览器有IE9+、Firefox3.5+、Chrome4+、Safari3+、Opera10+、iOS Mobile、Safari4.2+；

2、OpenType(.otf)格式

.otf字体被认为是一种原始的字体格式，其内置在TureType的基础上，支持这种字体的浏览器有Firefox3.5+、Chrome4.0+、Safari3.1+、Opera10.0+、iOS Mobile、Safari4.2+；

3、Web Open Font Format(.woff)格式

woff字体是Web字体中最佳格式，他是一个开放的TrueType/OpenType的压缩版本，同时也支持元数据包的分离，支持这种字体的浏览器有IE9+、Firefox3.5+、Chrome6+、Safari3.6+、Opera11.1+；

4、Embedded Open Type(.eot)格式

.eot字体是IE专用字体，可以从TrueType创建此格式字体，支持这种字体的浏览器有IE4+；

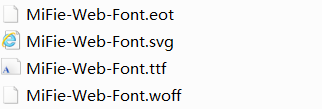
5、SVG(.svg)格式

.svg字体是基于SVG字体渲染的一种格式，支持这种字体的浏览器有Chrome4+、Safari3.1+、Opera10.0+、iOS Mobile Safari3.2+

1. 字体图标

常见的是把网页常用的一些小的图标，借助工具帮我们生成一个字体包，然后就可以像使用文字一样使用图标了

* + - 1. 优点
         * 将所有图标打包成字体库，减少请求；
         * 具有矢量性，可保证清晰度；
         * 使用灵活，便于维护
      2. 生成字体图标文件
         1. 下载字体图标文件：如https://www.awesomes.cn/的网站的介绍和使用



* + - * 1. 自定义字体

/\*定义字体图标\*/  
@font-face {  
 font-family: 'wjs'; //自定义的字体名称  
 src: url('../fonts/MiFie-Web-Font.eot'); /\* IE9\*/  
 src: url('../fonts/MiFie-Web-Font.eot') format('embedded-opentype'), /\* IE6-IE8 \*/  
 url('../fonts/MiFie-Web-Font.woff') format('woff'), /\* chrome、firefox \*/  
 url('../fonts/MiFie-Web-Font.ttf') format('truetype'), /\* chrome、firefox、opera、Safari, Android, iOS 4.2+\*/  
 url('../fonts/MiFie-Web-Font.svg') format('svg'); /\* iOS 4.1- \*/  
}

* + - * 1. 通过css样式使用字体

/\*自定义字体图标\*/  
.wjs\_font\_icon{  
 font-family: wjs; //这里对应着自定义的字体名称  
}  
/\*手机图标对应的编码\*/  
.wjs\_font\_icon\_phone::before{  
 content: "\e908"; //指定显示的内容  
}