# 1 颜色

新增了RGBA、HSLA模式，其中的A 表示透明度通道，即可以设置颜色值的透明度，相较opacity，它们不具有继承性，即不会影响子元素的透明度。

**RGBA是代表Red（红色） Green（绿色） Blue（蓝色）和 Alpha的[色彩空间](http://baike.baidu.com/view/974298.htm" \t "http://baike.baidu.com/_blank)。**

rgba(255,0,0,0.1) R、G、B 取值范围0~255

H 色调 取值范围0~360，0/360表示红色、120表示绿色、240表示蓝色

S 饱和度 取值范围0%~100%

L 亮度 取值范围 0%~100%

A 透明度 取值范围 0~1

关于透明度：



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

background-color: transparent;

3、使用rgba 来控制颜色，相对opacity ，不具有继承性

# 2 文本新增属性

## 2.1 文本阴影

text-shadow，可分别设置偏移量、模糊度、颜色（可设透明度）。

**text-shadow**: 3**px** 4**px** 5**px #ccc**;

3px 水平偏移量. 正值向右 负值向左  
 4px 垂直偏移量. 正值向下 负值向上  
 5px 模糊度 模糊度不能为负值，值越大越模糊  
 #ccc 设置对象阴影的颜色.

阴影叠加：text-shadow:2px 2px 0px red, 2px 2px 4px green;

先渲染后面的，再渲染前面的

几个好玩的例子：

层叠：color:red; font-size:100px; font-weight:bold; text-shadow:2px 2px 0px white, 4px 4px 0px red;

光晕：color:white; font-size:100px; text-shadow:0 0 10px #fff, 0 0 20px #fff, 0 0 30px #fff, 0 0 40px #ff00de, 0 0 70px #ff00de, 0 0 80px #ff00de, 0 0 100px #ff00de, 0 0 150px #ff00de;

火焰文字：text-shadow: 0 0 20px #fefcc9, 10px -10px 30px #feec85, -20px -20px 40px #ffae34, 20px -40px 50px #ec760c, -20px -60px 60px #cd4606, 0 -80px 70px #973716, 10px -90px 80px #451b0e; font-family:Verdana, Geneva, sans-serif; font-size:100px; font-weight:bold; color:white;

## 2.2 文字描边

webkit-text-stroke:宽度 颜色 （只有webkit内核浏览器才支持文字描边）

## 2.3 文字排版

direction:rtl;unicode-bidi:bidi-override;

定义文字排列方式(全兼容),rtl 从右向左排列,ltr 从右向左排列

注意要配合unicode-bidi 一块使用，才能实现排序，没有这个属性，那么文字只会走到右边或者左边，而不会改变顺序。

## 2.4 文字溢出

text-overflow 定义省略文本的处理方式

clip 无省略号

ellipsis 省略号 (注意配合overflow:hidden和white-space:nowrap一块使用)

## 2.5 文字字体

@font-face {

font-family: ‘myziti’; 字体名

src: url('111-webfont.eot');

src: url('111-webfont.eot?#iefix') format('embedded-opentype'),

url('111-webfont.woff') format('woff'),

url('111-webfont.ttf') format('truetype'),

url('111-webfont.svg#untitledregular') format('svg');

font-weight: normal;

font-style: normal;

}

使用：

p { font-family:‘myziti’}

转换字体格式生成兼容代码

<http://www.fontsquirrel.com/fontface/generator>

# 3 边框新增属性

## 3.1 边框圆角

border-radius: 1-4个数字 / 1-4个数字

前面是水平轴半径，后面是垂直轴半径，不给“/”则水平和垂直一样（即正圆圆角）。

如：border-radius: 10px/5px;值的单位可以是px，%等。

用法

1个：都一样

border-radius: 一样

2个：对角

border-radius: 左上&右下 右上&左下

3个：斜对角

border-radius: 左上 右上&左下 右下

4个：全部，顺时针

border-radius: 左上 右上 右下 左下

|  |
| --- |
| **border-radius**: 60**px**; */\* 四个角都是60 \*/* **border-top-left-radius**: 60**px**; */\* 左上角角都是60，同理还有右上角，右下角等 \*/* **border-radius**: 60**px** 30**px** 90**px** 120**px**; */\* 前两者是左上，右上半径，后两者是右下，左下半径 \*/* **border-radius**: 60**px**/30**px**; */\* 分别为左上，右下，剩余没写的取对角值 \*/* **border-radius**: 20**px** 60**px** 120**px**; */\* 依然是顺时针，左上，右上，右下，左下没写为其对角的值 \*/*  **border-radius**: 200**px** 000; */\** ***扇形****：200px是扇形所在矩形的宽高值 \*/* |

圆形：

|  |
| --- |
| .**box**{  **width**: 100**px**;  **height**: 100**px**;  **border**: 1**px solid #000**;  **border-radius**: 50%; } |

横放椭圆：

|  |
| --- |
| .**box**{  **width**: 100**px**;  **height**: 50**px**;  **border**: 1**px solid #000**;  **border-radius**: 100**px**/50**px**; } |

竖放椭圆：

|  |
| --- |
| .**box**{  **width**: 50**px**;  **height**: 100**px**;  **border**: 1**px solid #000**;  **border-radius**: 50%; } |

## 3.1 边框圆角案例-大风车

<style>  
 .box {  
 width: 400px;  
 height: 400px;  
 margin: 50px auto;  
 transition: 5s linear;  
 }  
  
 .box div {  
 width: 180px;  
 height: 180px;  
 margin: 10px;  
 border: 1px solid #5cb85c;  
 box-sizing: border-box;  
 float: left;  
 background-color: greenyellow;  
 }  
  
 .box div:nth-child(1), .box div:nth-child(4) {  
 border-radius: 0 70%;  
 }  
  
 .box div:nth-child(2), .box div:nth-child(3) {  
 border-radius: 70% 0;  
 }  
  
 .box:hover {  
 -webkit-transform: rotate(720deg);  
 }  
 </style>  
<div class="box">  
 <div></div>  
 <div></div>  
 <div></div>  
 <div></div>  
</div>

## 3.2 边框背景

border-image 设置边框的背景图片

border-image-source:url(“”) 设置边框图片的地址

border-image-slice:27,27,27,27 浏览器会自动去裁剪图片.

border-left-width:20px; 指定边框的宽度

border-image-repeat: stretch; 边框平铺的样式

stretch 拉升

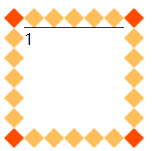
round 会自动调整缩放比例

repeat 重复

## 3.2 边框背景案例-切割图

<style>  
 .box {  
 width: 100px;  
 height: 100px;  
 border: 20px solid transparent;  
 border-image: url(border.png) 27 27 round round;  
 }  
 </style>  
<div class="box">1</div>

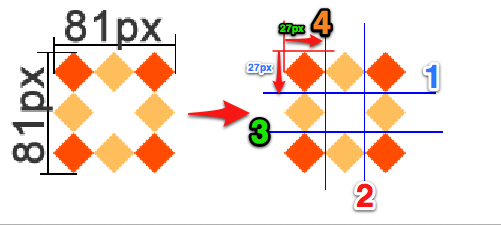
背景图： 效果图：

分析： 0 即是边框背景图的上下裁剪尺寸，10是左右裁剪尺寸，如果裁剪了10像素，而边框本身的尺寸没有达到裁剪的大小，那么裁剪图片会被相应的伸缩。

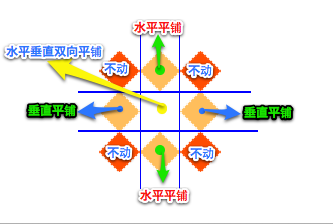
注意：切割图片数值不加 px单位

设置的图片将会被“切割”成九宫格形式，然后进行设置。如下图



“切割”完成后生成虚拟的9块图形，然后按对应位置设置背景，

其中四个角位置、形状保持不变，中心位置水平垂直两个方向平铺。如下图



**1、round和repeat之间的区别**

round 会自动调整尺寸，完整显示边框图片。

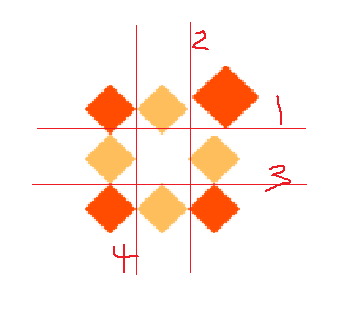


repeat 单纯平铺多余部分，会被“裁切”而不能完整显示。



**2、更改裁切尺寸**

background-slice: 34 36 27 27 分别设置裁切如下图





关于边框图片重点理解9宫格的裁切及平铺方式，实际开发中应用不广泛，但是如能灵活动用会给我们带来不少便利。

## 3.3 边框阴影

box-shadow 与 text-shadow 用法差不多

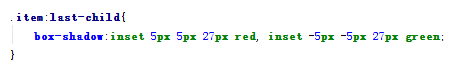
1、水平偏移量 正值向右 负值向左；

2、垂直偏移量 正值向下 负值向上；

*box-shadow: 5px 5px 27px red, -5px -5px 27px green;*

3、模糊度是不能为负值；

4、inset可以设置内阴影；



设置边框阴影不会改变盒子的大小，即不会影响其兄弟元素的布局。

可以设置多重边框阴影，实现更好的效果，增强立体感。

# 4 背景渐变

## 4.1 线性渐变(gradient变化)

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

从黄色渐变到绿色 

background:linear-gradient(

to right 表示方向 (left,top,right,left ,也可以使用度数)

yellow, 渐变起始颜色

green 渐变终止颜色

)

不写渐变方向，默认从上往下渐变

.box {  
 width: 500px;  
 height: 200px;  
 background:linear-gradient(30deg,green,orange);  
}

渐变也可以指定位置：到了指定位置如何变化。例如：（位置也可以写百分比）

.box {  
 width: 500px;  
 height: 100px;  
 background-image: linear-gradient(to right,green 30px,orange 100px);  
}

注意：使用background-image 也可以

## 4.1 线性渐变案例-进度条

<style>  
 .box {  
 width: 500px;  
 height: 100px;  
 background: linear-gradient(  
 135deg,  
 black 25%,  
 transparent 25%,  
 transparent 50%,  
 black 50%,  
 black 75%,  
 transparent 75%  
 );  
 background-size: 100px 100px;  
 animation: move 1s linear infinite;  
 }  
  
 @keyframes move {  
 from {}  
 to {  
 background-position: 100px 0;  
 }  
 }  
</style>

## 4.2 径向渐变 (radial径向)

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**



|  |
| --- |
| **background**: **radial-gradient**(  150**px at center**,  **yellow**,  **green** );  围绕中心点做渐变，半径是150px，从黄颜色到绿颜色做渐变. |

**1、必要的元素：**

a、辐射范围即圆半径 (半径越大，渐变效果越大)

b、中心点 即圆的中心 (中心点的位置是以盒子自身)

以左上角为圆的中心点

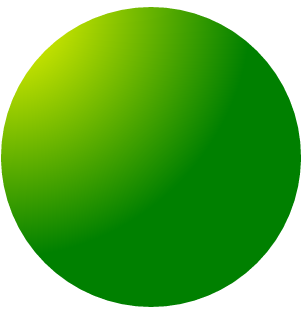
**background**: **radial-gradient**(  
 150**px at 0px 0px**,  
 **yellow**,  
 **green** );

c、渐变起始色

d、渐变终止色

**2、关于中心点：**中心位置参照的是盒子的左上角

**3、关于辐射范围：**其半径可以不等，即可以是椭圆



**div**{  
 **width**: 300**px**;  
 **height**: 300**px**;  
 **margin**:100**px auto**;  
 **background**: **radial-gradient**(  
 250**px at** 0**px** 0**px**,  
 **yellow**,  
 **green** );  
 **border-radius**: 150**px**;  
 }

# 5 背景新增属性

## 5.1 控制背景

background-size: 500px 500px; 控制背景大小，也可以是%

background-size:cover; 完全覆盖盒子，但不能保证是否完整显示

background-size:contain 背景图最大化在盒子中等比例显示，不保证铺满

cover会自动调整缩放比例，保证图片始终填充满背景区域，如有溢出部分则会被隐藏。

整个背景图片完整显示在背景区域.

contain会自动调整缩放比例，保证图片始终完整显示在背景区域。

注意：背景图默认从padding就开始平铺了，为了让背景从内容盒子才开始平铺显示，可以设置背景原点，backgound-origin:padding-box，将默认值改为：content-box

background-origin ： border | padding | content

border-box： 从border区域开始显示背景。

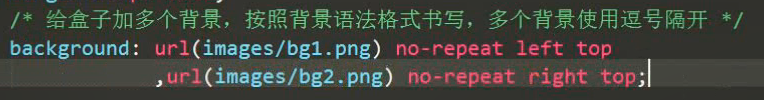
padding-box： 从padding区域开始显示背景。

content-box： 从content区域开始显示背景。

## 5.2 背景剪裁

background-clip: padding-box 超出padding-box的裁剪，同理有border-box，content-box

## 5.3 多背景



# 6 缩放

缩放属性：resize:both;overflow:auto;

both 水平垂直都可以缩放

horizontal 只有水平方向可以缩放

vertical 只有垂直方向可以缩放

注意：一定要配合overflow:auto 一块使用

# 7 遮罩

<style>  
 .box {  
 width: 800px;  
 height: 600px;  
 background: url(\*\*\*.jpg) no-repeat;  
 background-size: 100% 100%;  
 border: 10px solid #000;  
 -webkit-mask: url(mask.png) no-repeat;  
 transition: 1s;  
 }  
  
 .box:hover {  
 -webkit-mask-position: 100% 100%;  
 }  
</style>