# 目录

[目录 2](#_Toc4353)

[一、行高和字号 3](#_Toc426)

[1.1 行高 3](#_Toc19877)

[1.2 单行文本垂直居中 3](#_Toc13188)

[1.3 font属性 4](#_Toc31075)

[二、超级链接的美化 6](#_Toc635)

[2.1 伪类 6](#_Toc21384)

[2.2 超级链接的美化 7](#_Toc26136)

[三、background系列属性 8](#_Toc32051)

[3.1 background-color属性 8](#_Toc27570)

[3.1.1 用英语单词来表示 8](#_Toc25648)

[3.1.2 用rgb方法来表示 8](#_Toc20593)

[3.1.3 十六进制表示法 9](#_Toc25969)

[3.2 background-image 12](#_Toc18522)

[3.3 background-repeat属性 13](#_Toc27730)

[3.4 background-position属性 14](#_Toc17155)

[3.4.1 属性的意思 14](#_Toc15638)

[3.4.2 css精灵 15](#_Toc17051)

[3.4.3 用单词描述 16](#_Toc19521)

[3.5 background-attachment 18](#_Toc16483)

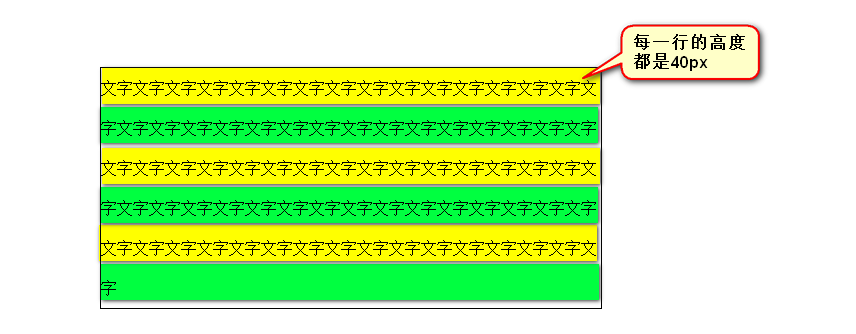
[3.6 background综合属性 18](#_Toc20914)

# 一、行高和字号

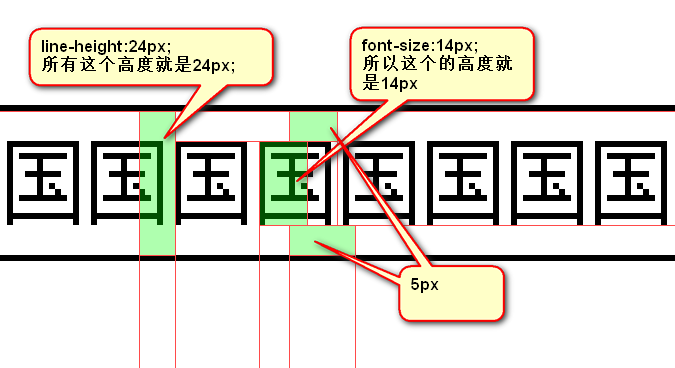
## 1.1 行高

CSS中，所有的行，都有行高。盒模型的padding，绝对不是直接作用在文字上的，而是作用在“行”上的。

|  |
| --- |
| 1. line-height: 40px; |



文字，是在自己的行里面居中的。比如，现在文字字号14px，行高是24px。那么：



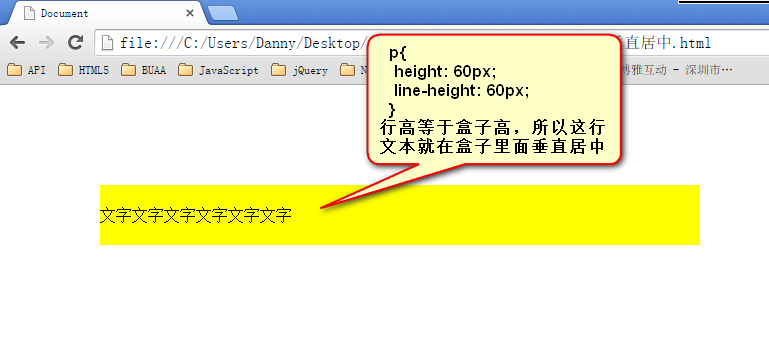
为了严格保证字在行里面居中，我们的工程师有一个约定： **行高、字号，一般都是偶数。**这样，它们的差，就是偶数，就能够被2整除。

## 1.2 单行文本垂直居中

文本在行里面居中

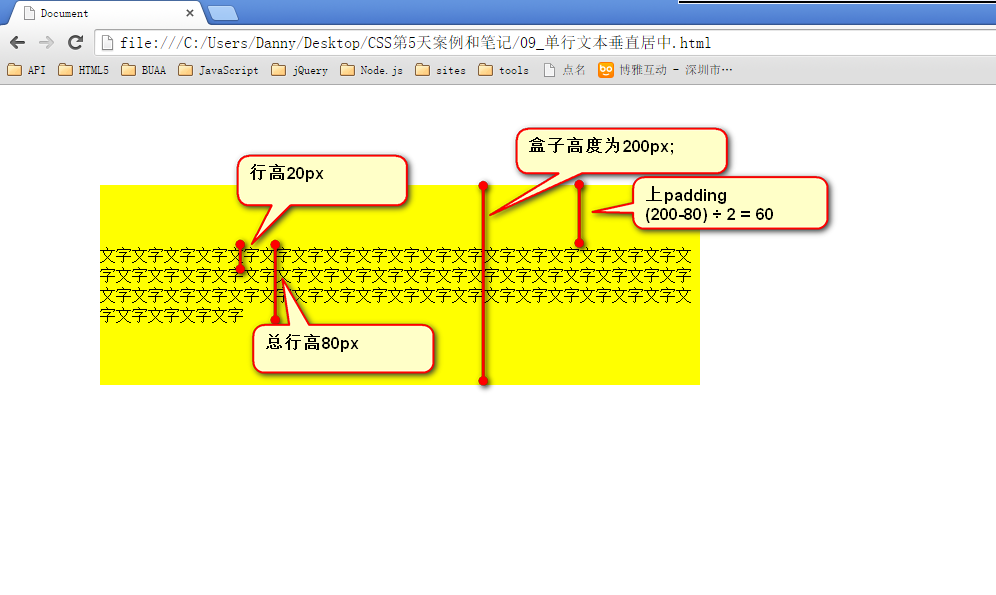
公式：

|  |
| --- |
| 1. 行高:盒子高; |



需要注意的是，这个小技巧，行高=盒子高。 **只适用于单行文本垂直居中！！不适用于多行**。

**如果想让多行文本垂直居中，需要设置盒子的padding**：



## 1.3 font属性

**● 使用font属性，能够将字号、行高、字体，能够一起设置**。

|  |
| --- |
| 1. **font: 14px/24px “宋体”;** |

等价于三行语句：

|  |
| --- |
| 1. font-size:14px; 2. line-height:24px; 3. font-family:"宋体"; |

font-family就是“字体”。family是“家庭”、“伐木累”的意思。

**● 网页中不是所有字体都能用哦，因为这个字体要看用户的电脑里面装没装**，比如你设置：

|  |
| --- |
| 1. font-family: "华文彩云"; |

如果用户电脑里面没有这个字体，那么就会变成宋体。

**页面中，中文我们只使用： 微软雅黑、宋体、黑体。 如果页面中，需要其他的字体，那么需要切图。**

**英语：Arial 、 Times New Roman**

**● 为了防止用户电脑里面，没有微软雅黑这个字体。就要用英语的逗号，隔开备选字体**，就是说如果用户电脑里面，没有安装微软雅黑字体，那么就是宋体：

|  |
| --- |
| 1. font-family: "微软雅黑","宋体"; |

备选字体可以有无数个，用逗号隔开。

**● 我们要将英语字体，放在最前面，这样所有的中文，就不能匹配英语字体，就自动的变为后面的中文字体**：

|  |
| --- |
| 1. font-family: "Times New Roman","微软雅黑","宋体"; |

**● 所有的中文字体，都有英语别名，我们也要知道**：

微软雅黑的英语别名：

|  |
| --- |
| 1. font-family: "Microsoft YaHei"; |

宋体的英语别名：

|  |
| --- |
| 1. font-family: "SimSun"; |

font属性能够将font-size、line-height、font-family合三为一：

|  |
| --- |
| 1. font:12px/30px "Times New Roman"**,**"Microsoft YaHei"**,**"SimSun"; |

**● 行高可以用百分比，表示字号的百分之多少**。一般来说，都是大于100%的，因为行高一定要大于字号。

|  |
| --- |
| 1. font:12px/**200%** “宋体” |

等价于

|  |
| --- |
| 1. font:12px/**24px** “宋体”; |

反过来，比如：

|  |
| --- |
| 1. font:16px/**48px** “宋体”; |

等价于

|  |
| --- |
| 1. font:16px/**300%** “宋体” |

# 二、超级链接的美化

超级链接就是a标签。

## 2.1 伪类

也就是说，同一个标签，根据用户的某种状态不同，有不同的样式。这就叫做“伪类”。

类就是工程师加的，比如div属于box类，很明确，就是属于box类。但是a属于什么类？不明确。因为要看用户有没有点击、有没有触碰。所以，就叫做“伪类”。

伪类用冒号来表示。

a标签有4种伪类，要求背诵：

|  |
| --- |
| 1. a**:link**{ 2. color:red; 3. } 4. a**:visited**{ 5. color:orange; 6. } 7. a**:hover**{ 8. color:green; 9. } 10. a**:active**{ 11. color:black; 12. } |

:link 表示， 用户没有点击过这个链接的样式。 是英语“链接”的意思。

:visited 表示， 用户访问过了这个链接的样式。 是英语“访问过的”的意思。

:hover 表示， 用户鼠标悬停的时候链接的样式。 是英语“悬停”的意思。

:active 表示， 用户用鼠标点击这个链接，但是不松手，此刻的样式。 是英语“激活”的意思。

|  |  |  |  |
| --- | --- | --- | --- |
| a:link | a:visited | a:hover | a:active |

记住，这四种状态，在css中，必须按照固定的顺序写：

a:**l**ink 、a:**v**isited 、a:**h**over 、a:**a**ctive

如果不按照顺序，那么将失效。“爱恨准则”**l**o**v**e **ha**te。必须先爱，后恨。

## 2.2 超级链接的美化

a标签在使用的时候，非常的难。因为不仅仅要控制a这个盒子，也要控制它的伪类。

我们一定要将a标签写在前面，:link、:visited、:hover、:active这些伪类写在后面。

**a标签中，描述盒子； 伪类中描述文字的样式、背景。**

|  |
| --- |
| 1. .nav ul li **a**{ 2. display: block; 3. width: 120px; 4. height: 40px; 5. } 6. .nav ul li **a:link** ,.nav ul li a**:visited**{ 7. text-decoration: none; 8. background-color: yellowgreen; 9. color:white; 10. } 11. .nav ul li **a:hover**{ 12. background-color: purple; 13. font-weight: bold; 14. color:yellow; 15. } |

记住，所有的a不继承text、font这些东西。因为a自己有一个伪类的权重。

最标准的，就是把link、visited、hover都要写。但是前端开发工程师在大量的实践中，发现不写link、visited浏览器也挺兼容。所以这些“老油条”们，就把a标签简化了：

a:link、a:visited都是可以省略的，简写在a标签里面。也就是说，a标签涵盖了link、visited的状态。

|  |
| --- |
| 1. .nav ul li a{ 2. display: block; 3. width: 120px; 4. height: 50px; 5. text-decoration: none; 6. background-color: purple; 7. color:white; 8. } 9. .nav ul li a:hover{ 10. background-color: orange; 11. } |

# 三、background系列属性

## 3.1 background-color属性

背景颜色属性。

css2.1中，颜色的表示方法有哪些？一共有三种：单词、rgb表示法、十六进制表示法

### 3.1.1 用英语单词来表示

能够用英语单词来表述的颜色，都是简单颜色。

红色：

|  |
| --- |
| 1. background-color: red; |

### 3.1.2 用rgb方法来表示

红色：

|  |
| --- |
| 1. background-color: **rgb(255,0,0)**; |

rgb表示三原色“红”red、“绿”green、“蓝”blue。光学显示器，每个像素都是由三原色的发光原件组成的，靠明亮度不同调成不同的颜色的。

**用逗号隔开，r、g、b的值，每个值的取值范围0~255，一共256个值。**

如果此项的值，是255，那么就说明是纯色：

绿色：

|  |
| --- |
| 1. background-color: rgb(0,255,0); |

蓝色：

|  |
| --- |
| 1. background-color: rgb(0,0,255); |

黑色：

|  |
| --- |
| 1. background-color: rgb(0,0,0); |

光学显示器，每个元件都不发光，黑色的。

白色：

|  |
| --- |
| 1. background-color: rgb(255,255,255); |

颜色可以叠加，比如黄色就是红色和绿色的叠加：

|  |
| --- |
| 1. background-color: rgb(255,255,0); |

再比如：

|  |
| --- |
| 1. background-color: rgb(111,222,123); |

就是红、绿、蓝三种颜色的不同比例叠加。

### 3.1.3 十六进制表示法

红色：

|  |
| --- |
| 1. background-color: **#ff0000**; |

所有用#开头的值，都是16进制的。

#ff0000

16进制表示法，也是两位两位看，看r、g、b，但是没有逗号隔开。

ff就是10进制的255 ，00 就是10进制的0，00就是10进制的0。所以等价于rgb(255,0,0);

怎么换算的？我们介绍一下

我们现在看一下10进制中的基本数字（一共10个）:

0、1、2、3、4、5、6、7、8、9

16进制中的基本数字（一共16个）:

0、1、2、3、4、5、6、7、8、9、a、b、c、d、e、f

16进制对应表：

十进制数 十六进制数

0 0

1 1

2 2

3 3

……

10 a

11 b

12 c

13 d

14 e

15 f

16 10

17 11

18 12

19 13

……

43 2b

……

255 ff

十六进制中，13 这个数字表示什么？

**表示1个16和3个1。 那就是19**。 这就是**位权**的概念，开头这位表示多少个16，末尾这位表示多少个1。

小练习：

16进制中28等于10进制多少？

答：2\*16+8 = 40。

16进制中的2b等于10进制多少？

答：2\*16+11 = 43。

16进制中的af等于10进制多少？

答：10 \* 16 + 15 = 175

16进制中的ff等于10进制多少？

答：15\*16 + 15 = 255

所以，#ff0000就等于rgb(255,0,0)

|  |
| --- |
| 1. background-color: #**123456**; |

等价于：

|  |
| --- |
| 1. background-color: rgb(**18**,**52**,**86**); |

所以，**任何一种十六进制表示法，都能够换算成为rgb表示法。也就是说，两个表示法的颜色数量，一样**。

**十六进制可以简化为3位，所有#aabbcc的形式，能够简化为#abc;**

比如：

|  |
| --- |
| 1. background-color:#ff0000; |

等价于

|  |
| --- |
| 1. background-color:#f00; |

比如：

|  |
| --- |
| 1. background-color:#112233; |

等价于

|  |
| --- |
| 1. background-color:#123; |

只能上面的方法简化，比如

|  |
| --- |
| 1. background-color:#222333; |

无法简化！

再比如

|  |
| --- |
| 1. background-color:#123123; |

无法简化！

要记住：

|  |
| --- |
| 1. #000 黑 2. #fff 白 3. #f00 红 4. #333 灰 5. #222 深灰 6. #ccc 浅灰 |

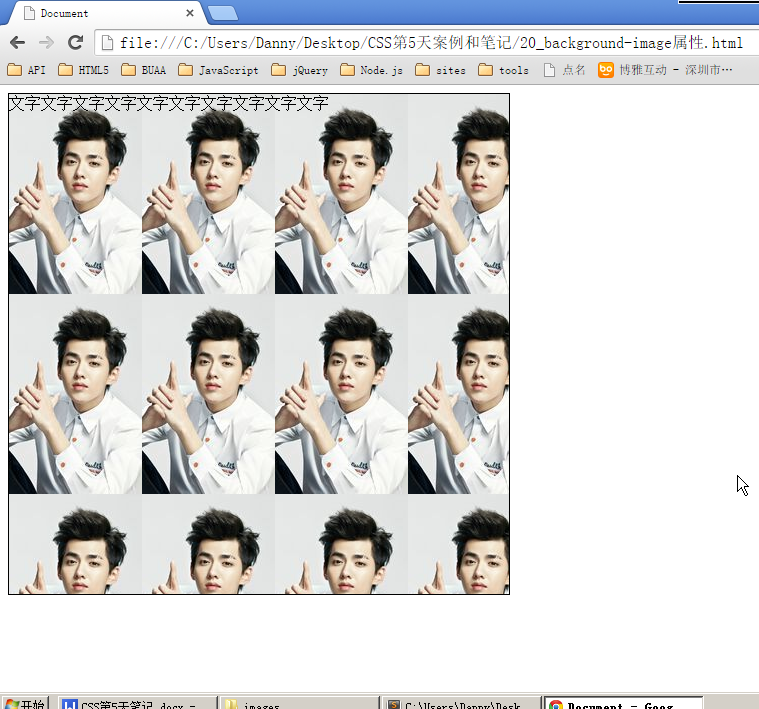
## 3.2 background-image

用于给盒子加上背景图片：

|  |
| --- |
| 1. background-image:**url(images/wuyifan.jpg)**; |

url()表示网址，uniform resouces locator 同意资源定位符

images/wuyifan.jpg 就是相对路径。



背景天生是会被平铺满的。

padding的区域有背景图。

## 3.3 background-repeat属性

设置背景图是否重复的，重复方式的。

repeat表示“重复”。

|  |  |  |  |
| --- | --- | --- | --- |
| 默认 | background-repeat:no-repeat; | background-repeat:repeat-x; | background-repeat:repeat-y; |

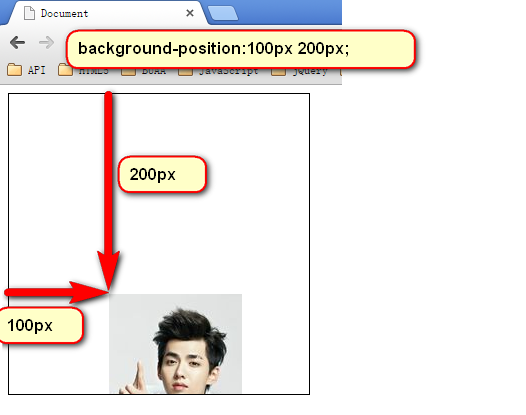
也就是说，background-repeat属性，有三种值：

|  |
| --- |
| 1. background-repeat:no-repeat; 不重复 2. background-repeat:repeat-x; 横向重复 3. background-repeat:repeat-y; 纵向重复 |

## 3.4 background-position属性

### 3.4.1 属性的意思

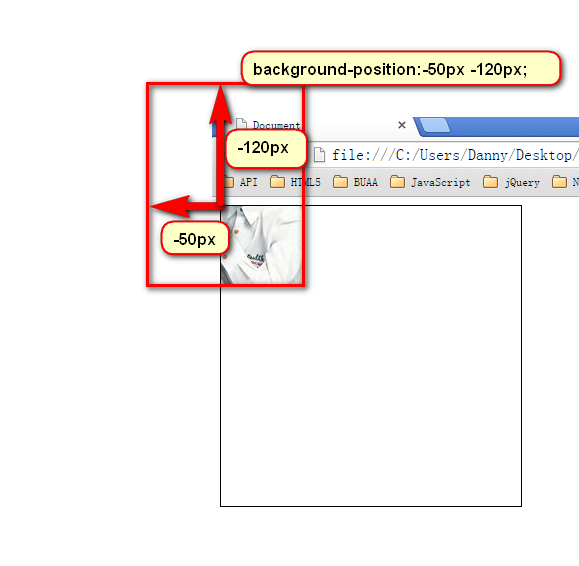
背景定位属性，是最难的属性。一定要好好学。



position就是“位置”的意思。background-position就是背景定位属性。

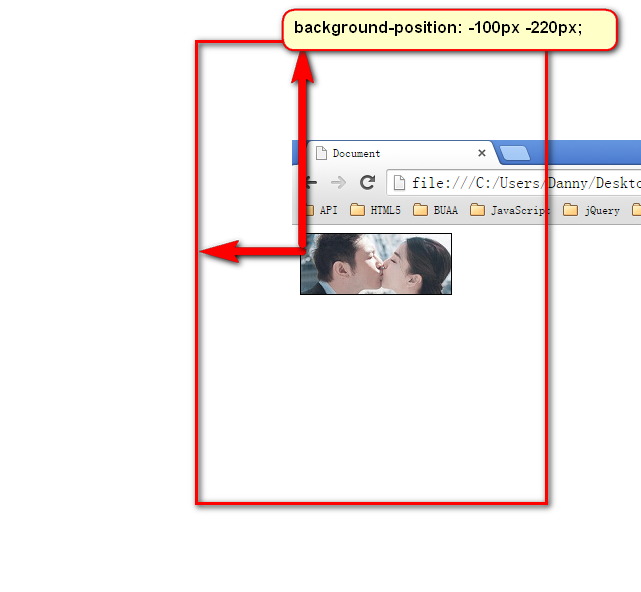
|  |
| --- |
| 1. background-position:向右移动量 向下移动量; |

定位属性可以是负数：



### 3.4.2 css精灵

原理：

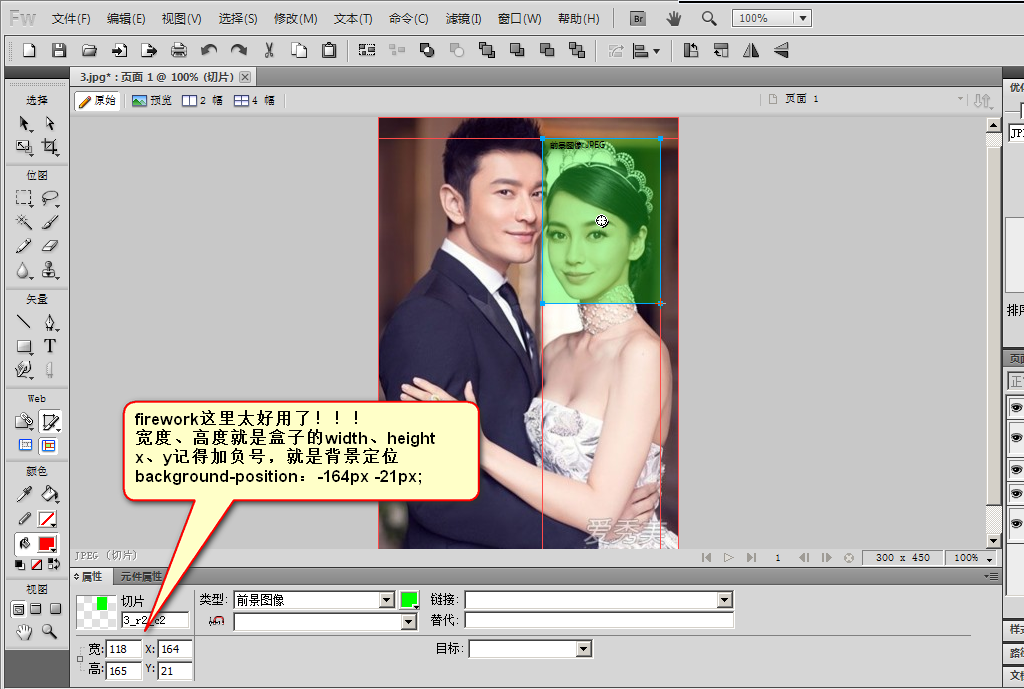


“css精灵”，英语css sprite，所以也叫做“css雪碧”技术。是一种CSS图像合并技术，该方法是将小图标和背景图像合并到一张图片上，然后利用css的背景定位来显示需要显示的图片部分。

css精灵有什么优点，就是减少了http请求。比如4张小图片，原本需要4个http请求。但是用了css精灵，小图片变为了一张图，http请求只有1个了。

淘宝网的精灵图：

用fireworks精确控制精灵：



### 3.4.3 用单词描述

|  |
| --- |
| 1. background-position: 描述左右的词儿 描述上下的词儿; |

描述左右的词儿： left、 center、right

描述上下的词儿： top 、center、bottom

所以：

|  |
| --- |
| 1. background-position: right bottom; |

右下角：

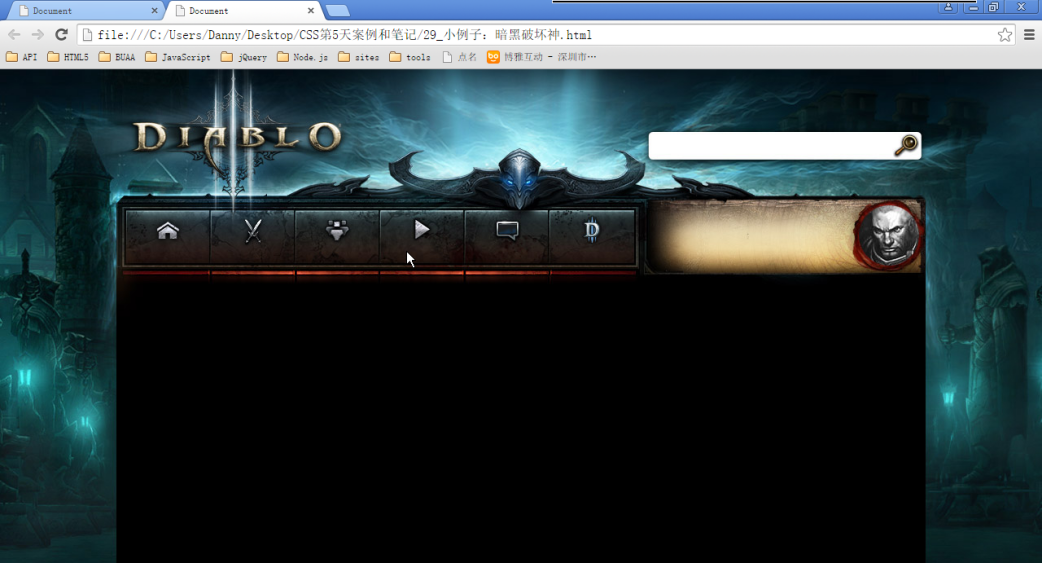


左下角：

|  |
| --- |
| 1. background-position: left bottom; |

用图：

1） 大背景图居中



2） 通栏banner



## 3.5 background-attachment

背景是否固定。

|  |
| --- |
| 1. background-attachment:fixed; |

背景就会被固定住，不会被滚动条滚走。

## 3.6 background综合属性

background属性和border一样，是一个综合属性：

|  |
| --- |
| 1. background:red url(1.jpg) no-repeat 100px 100px fixed; |

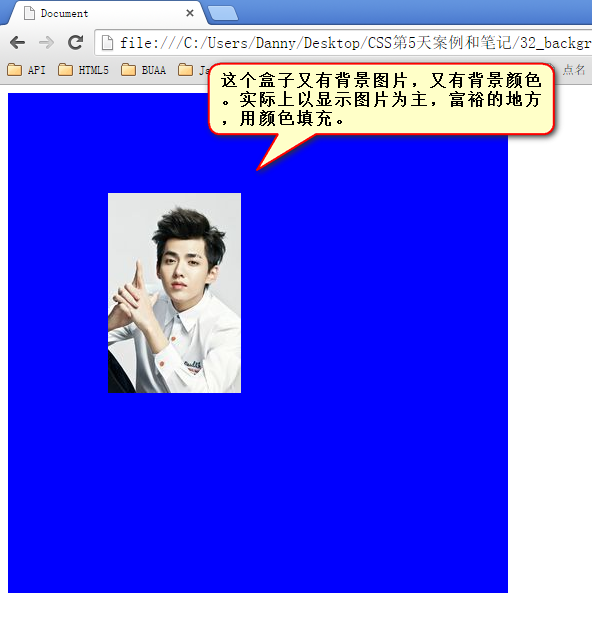
等价于：

|  |
| --- |
| 1. background-color:red; 2. background-image:url(1.jpg); 3. background-repeat:no-repeat; 4. background-position:100px 100px; 5. background-attachment:fixed; |

可以任意省略部分：

|  |
| --- |
| 1. background: red; |

|  |
| --- |
| 1. background: blue url(images/wuyifan.jpg) no-repeat 100px 100px; |



精灵的使用：

|  |
| --- |
| 1. background: url(images/taobao.png) no-repeat 0 -133px; |