# 目录

[目录 2](#_Toc31486)

[一、复习 3](#_Toc17033)

[二、相对定位 5](#_Toc4838)

[2.1 认识相对定位 5](#_Toc16130)

[2.2 不脱标，老家留坑，形影分离 6](#_Toc27779)

[2.3 相对定位用途 6](#_Toc29234)

[2.4 相对定位的定位值 6](#_Toc2947)

[三、绝对定位 8](#_Toc419)

[3.1 绝对定位脱标 8](#_Toc21878)

[3.2 参考点 9](#_Toc534)

[3.3 以盒子为参考点 11](#_Toc25567)

[3.4 绝对定位的盒子居中 13](#_Toc10165)

[四、固定定位 14](#_Toc16)

[五、z-index 15](#_Toc11546)

# 一、复习

a标签的伪类4个：

a:link 没有被点击过的链接

a:visited 访问过的链接

a:hover 悬停

a:active 按下鼠标不松手

顺序就是“**l**o**v**e **ha**te”准则。

可以简写：

|  |
| --- |
| 1. a{ 3. } 4. a:hover{ 6. } |

background系列属性，CSS2.1层面

|  |
| --- |
| 1. background-color 2. background-image 3. background-repeat 4. background-position 5. background-attachment |

background-color:red;

颜色表示法：red、rgb(255,0,0)、#**ff0000**、#f00

rgb和十六进制颜色是一样多的，是一一对应的。有换算关系。

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

默认是平铺的。

background-repeat:no-repeat;

重复横向： repeat-x;

重复纵向： repeat-y;

不重复： no-repeat;

background-position:100px 200px;

背景图在盒子中右边移动100px，向下移动200px。

可以是负数，background-position:-100px -200px;

向左边100px，向上边200px。

英语单词来表示

background-position:right bottom; 右下角

background-position:center bottom; 下边居中

css精灵

css雪碧，要会用fireworks精确显示精灵图。

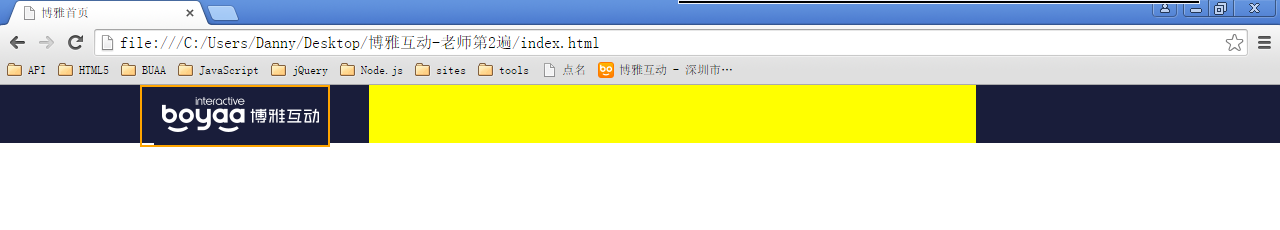
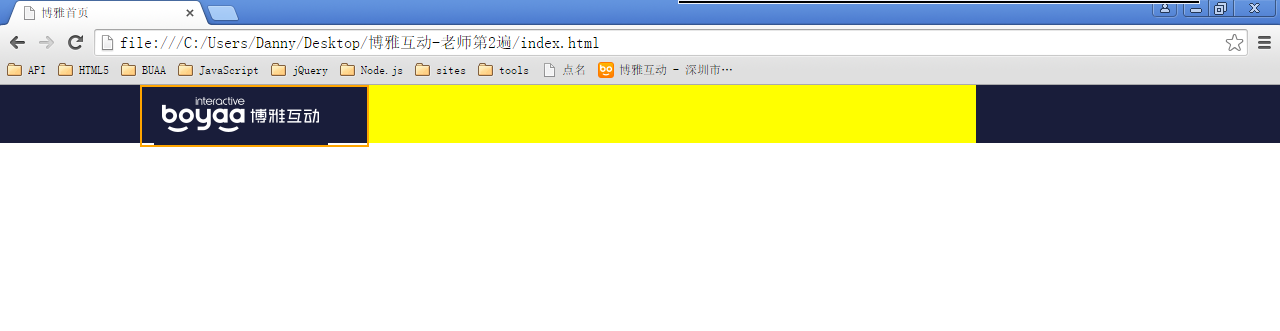
background-attachment

背景固定

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

background属性大综合

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



# 二、相对定位

定位有三种，分别是相对定位、绝对定位、固定定位。

相对定位：

|  |
| --- |
| 1. position:relative; |

绝对定位：

|  |
| --- |
| 1. position:absolute; |

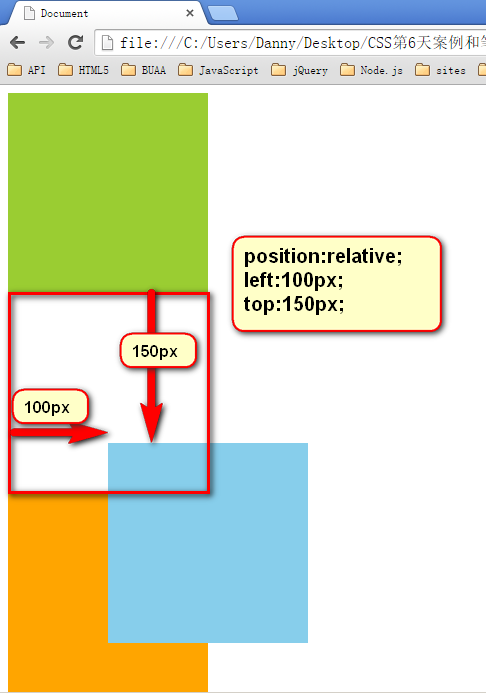
固定定位：

|  |
| --- |
| 1. position:fixed; |

每一种定位，都暗藏玄机，所以我们分别讲解。

## 2.1 认识相对定位

相对定位，就是微调元素位置的。让元素**相对自己原来的位置，进行位置调整**。



也就是说，如果一个盒子想进行位置调整，那么就要使用相对定位

|  |
| --- |
| 1. position:relative; → 必须先声明，自己要相对定位了， 2. left:100px; → 然后进行调整。 3. top:150px; → 然后进行调整。 |

## 2.2 不脱标，老家留坑，形影分离

相对定位不脱标，真实位置是在老家，只不过影子出去了，可以到处飘。



## 2.3 相对定位用途

相对定位有坑，所以一般不用于做“压盖”效果。页面中，效果极小。就两个作用：

1） **微调**元素

2） 做绝对定位的参考，子绝父相（讲绝对定位的时候说）

## 2.4 相对定位的定位值

可以用left、right来描述盒子右、左的移动；

可以用top、bottom来描述盒子的下、上的移动。

↘：

|  |
| --- |
| 1. position: relative; 2. top: 10px; 3. left: 40px; |

↙：

|  |
| --- |
| 1. position: relative; 2. **right: 100px; → 往左边移动** 3. top: 100px; |

↖：

|  |
| --- |
| 1. position: relative; 2. right: 100px; 3. **bottom: 100px; → 移动方向是向上。** |

↗：

|  |
| --- |
| 1. position: relative; 2. **top: -200px; → 负数就是相反的方向，如果是正，就是下边，如果是负数就是上边** 3. **right: -200px;** |

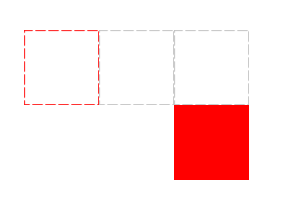
↗：

|  |
| --- |
| 1. position: relative; 2. **right: -300px;** 3. bottom: 300px; |

完全等价于：

|  |
| --- |
| 1. position: relative; 2. **left: 300px;** 3. bottom: 300px; |

如图，有几种相对定位的移动方法？



方法1：

|  |
| --- |
| 1. position:relative; 2. top:100px; 3. left:200px; |

方法2：

|  |
| --- |
| 1. position:relative; 2. bottom:-100px; 3. right:-200px; |

方法3：

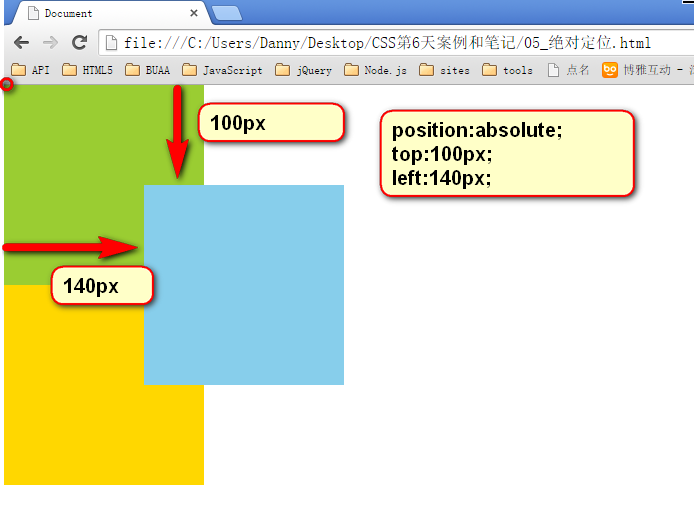
|  |
| --- |
| 1. position:relative; 2. top:100px; 3. right:-200px; |

方法4：

|  |
| --- |
| 1. position:relative; 2. bottom:-100px; 3. left:200px; |

# 三、绝对定位

绝对定位比相对定位更灵活。



## 3.1 绝对定位脱标

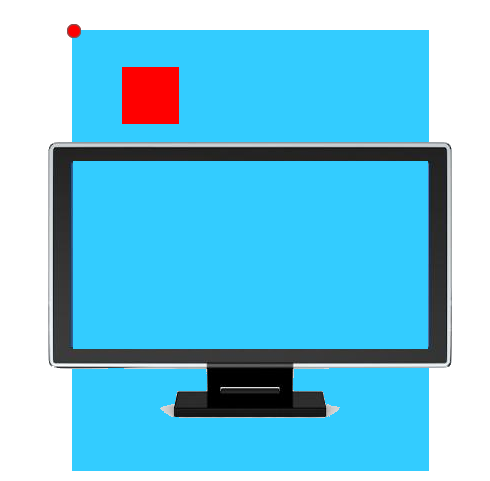
绝对定位的盒子，是脱离标准文档流的。所以，所有的标准文档流的性质，绝对定位之后都不遵守了。

绝对定位之后，标签就不区分所谓的行内元素、块级元素了，不需要display:block;就可以设置宽、高了：

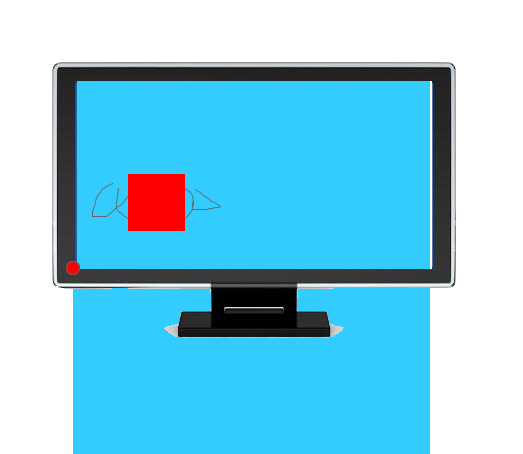
|  |
| --- |
| 1. span{ 2. **position: absolute;** 3. top: 100px; 4. left: 100px; 5. width: 100px; 6. height: 100px; 7. background-color: pink; 8. } |

## 3.2 参考点

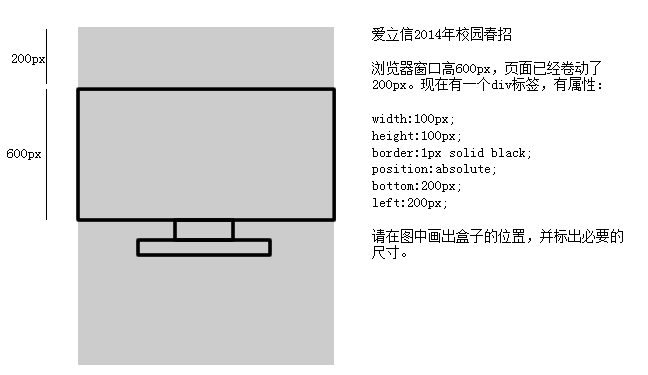
**绝对定位的参考点，如果用top描述，那么定位参考点就是页面的左上角，而不是浏览器的左上角：**



**如果用bottom描述，那么就是浏览器首屏窗口尺寸，对应的页面的左下角**：

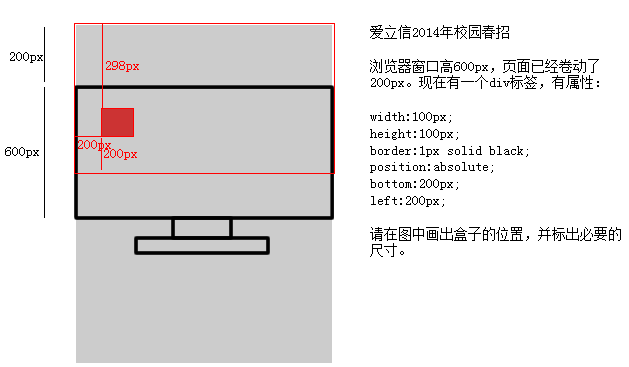


面试题：



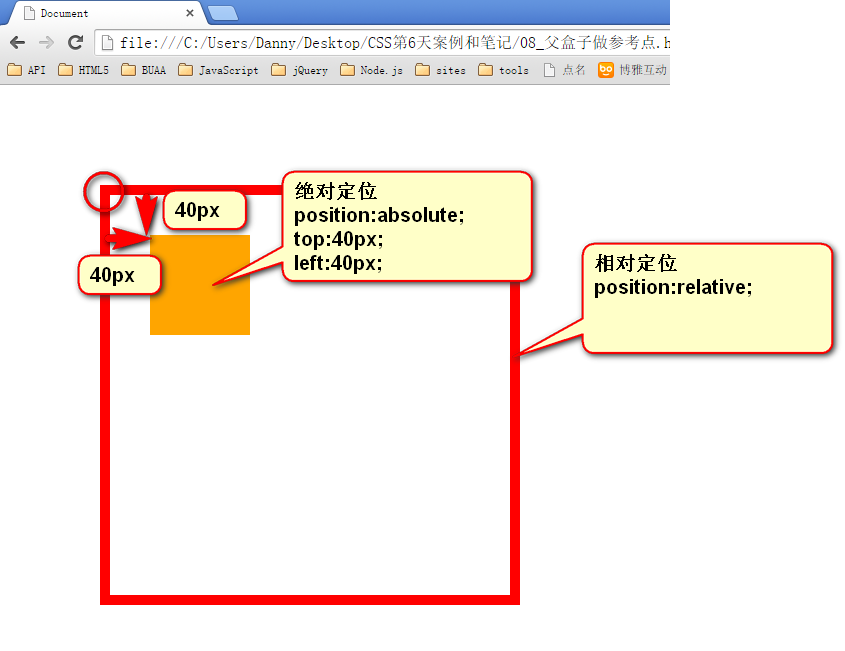
答案：

用bottom的定位的时候，参考的是浏览器首屏大小对应的页面左下角。



## 3.3 以盒子为参考点

一个绝对定位的元素，如果父辈元素中出现了也定位了的元素，那么将以父辈这个元素，为参考点。



● 要听最近的已经定位的祖先元素的，不一定是父亲，可能是爷爷：

|  |
| --- |
| 1. <div class="box1"> → 相对定位 2. <div class="box2"> → 没有定位 3. <p></p> → 绝对定位，将以box1为参考，因为box2没有定位，box1就是最近的父辈元素 4. </div> 5. </div> |

|  |
| --- |
| 1. <div class="box1"> → 相对定位 2. <div class="box2"> → 相对定位 3. <p></p> → 绝对定位，将以box2为参考，因为box2是自己最近的父辈元素 4. </div> 5. </div> |

● 不一定是相对定位，任何定位，都可以作为参考点

|  |
| --- |
| 1. <div> → 绝对定位 2. <p></p> → 绝对定位，将以div作为参考点。因为父亲定位了。 3. </div> |

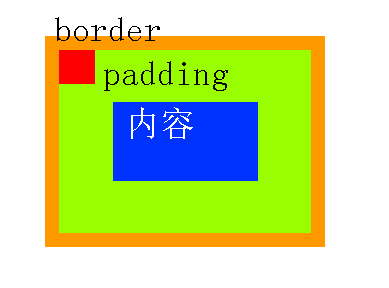
子绝父绝、子绝父相、子绝父固，都是可以给儿子定位的。但是，工程上子绝、父绝，没有一个盒子在标准流里面了，所以页面就不稳固，没有任何实战用途。工程上，“子绝父相”有意义，父亲没有脱标，儿子脱标在父亲的范围里面移动。

|  |
| --- |
| 1. <div class=”box1”> → 绝对定位 2. <div class=”box2”> → 相对定位 3. <div class=”box3”> → 没有定位 4. <p></p> → 绝对定位，以box2为参考定位。 5. </div> 6. </div> 7. </div> |

● 绝对定位的儿子，无视参考的那个盒子的padding。

下图中，绿色部分是div的padding，蓝色部分是div的内容区域。那么此时，div相对定位，p绝对定位。

p将无视父亲的padding，在border内侧为参考点，进行定位：



## 3.4 绝对定位的盒子居中

绝对定位之后，所有标准流的规则，都不适用了。所以margin:0 auto;失效。

|  |  |
| --- | --- |
|  |  |

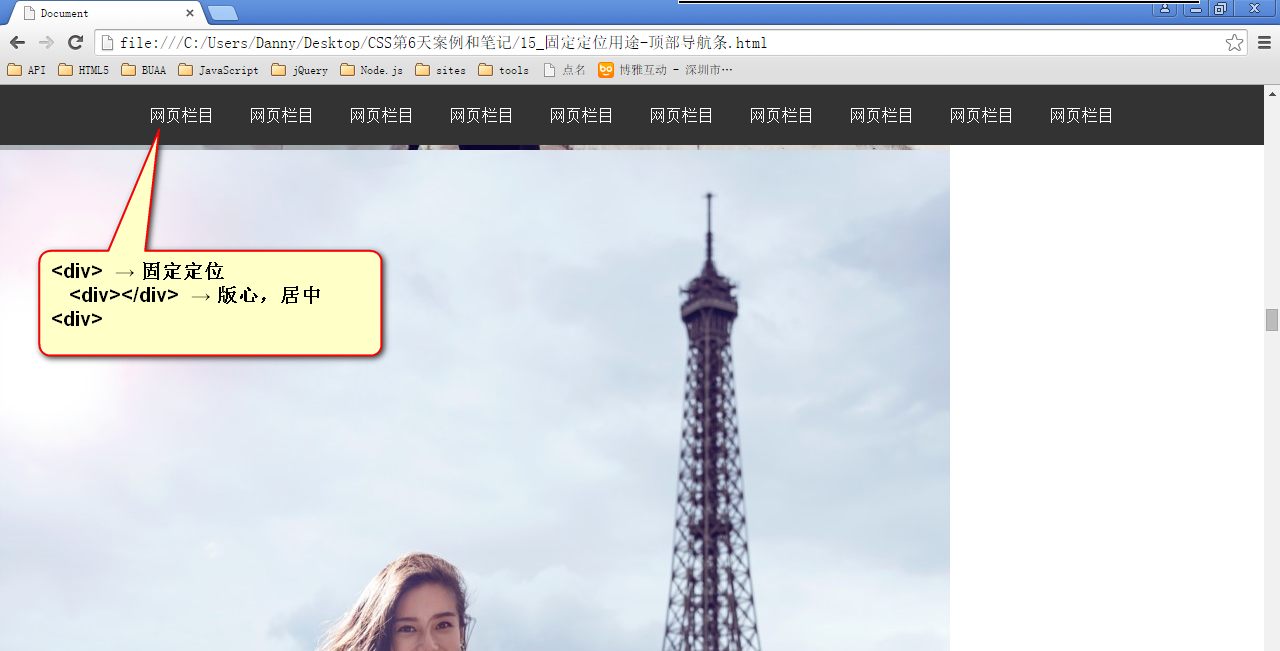
|  |
| --- |
| 1. width: 600px; 2. height: 60px;   **position: absolute;**   1. **left: 50%;** 2. top: 0; 3. **margin-left: -300px; → 宽度的一半** |

非常简单，当做公式记忆下来。就是left:50%; margin-left:负的宽度的一半。

# 四、固定定位

固定定位，就是相对浏览器窗口定位。页面如何滚动，这个盒子显示的位置不变。

固定定位脱标！



**IE6不兼容。**

# 五、z-index

● z-index值表示谁压着谁。数值大的压盖住数值小的。

● 只有定位了的元素，才能有z-index值。也就是说，不管相对定位、绝对定位、固定定位，都可以使用z-index值。而浮动的东西不能用。

● z-index值没有单位，就是一个正整数。默认的z-index值是0。

● 如果大家都没有z-index值，或者z-index值一样，那么谁写在HTML后面，谁在上面能压住别人。定位了的元素，永远能够压住没有定位的元素。

**● 从父现象：父亲怂了，儿子再牛逼也没用。**

**没有单位：**

|  |
| --- |
| 1. z-index: 988; |

