**CSS五种定位方式详解**

**定位**

**1、定位的组成**

所谓定位，简单的来说就是将盒子定在某一个位置。所以定位其实也是按照规则进行摆盒子，其规则为：**定位 = 定位模式 + 边偏移** 。

**1.1、定位模式**

定位模式决定元素的定位方式 ，它通过 CSS 的 **position 属性**来设置，其值可以分为五个：

| **值** | **描述** |
| --- | --- |
| absolute | 生成绝对定位的元素，相对于 static 定位以外的第一个父元素进行定位。元素的位置通过 “left”, “top”, “right” 以及 “bottom” 属性进行规定。 |
| fixed | 生成固定定位的元素，相对于浏览器窗口进行定位。元素的位置通过 “left”, “top”, “right” 以及 “bottom” 属性进行规定。 |
| relative | 生成相对定位的元素，相对于其正常位置进行定位。因此，“left:20” 会向元素的 LEFT 位置添加 20 像素。 |
| static | 默认值。没有定位，元素出现在正常的流中（忽略 top, bottom, left, right 或者 z-index 声明） |
| sticky | 粘性定位。根据用户的滚动位置进行定位。 |

**1.2、偏移量**

**元素具备了定位属性之后，可以使用的顶部top，底部bottom，左侧left和右侧right属性定位(static除外)**。他们也有不同的工作方式，这取决于定位方法。

| **属性** | **说明** |
| --- | --- |
| position | 指定元素的定位类型。 |
| right | 定义了定位元素右外边距边界与其包含块右边界之间的偏移。 |
| top | 定义了一个定位元素的上外边距边界与其包含块上边界之间的偏移。 |
| bottom | 定义了定位元素下外边距边界与其包含块下边界之间的偏移。 |
| left | 定义了定位元素左外边距边界与其包含块左边界之间的偏移。 |

**什么是文档流？**

将窗体自上而下分成一行行, 并在每行中按从左至右的顺序排放元素,即为文档流。

只有三种情况会使得元素脱离文档流，分别是：浮动、绝对定位和固定定位。

**2、静态定位 static(默认)**

HTML 元素默认情况下的定位方式为 static（静态）。

静态定位的元素不受 top、bottom、left 和 right 属性的影响。

position: static; 的元素不会以任何特殊方式定位；它始终根据页面的正常流进行定位。

**3、相对定位 relative(重点)**

relative定位，又称为相对定位，从字面上来解析，我们就可以看出该属性的主要特性：相对。但是它相对的又是相对于什么地方而言的呢？这个是个重点，也是最让人迷糊的一个地方，现在让我们来做个测试，我想大家都会明白的：

**1、初始未定位**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<style type="text/css">

#first {

width: 200px;

height: 100px;

border: 1px solid red;

}

#second {

width: 200px;

height: 100px;

border: 1px solid blue;

}

</style>

</head>

<body>

<div id="first">first</div>

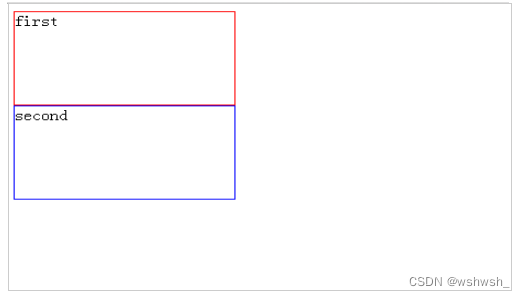
<div id="second">second</div>

</body>

</html>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png

初始原图：



**2、我们修改first元素的position属性：**

<style type="text/css">

#first {

width: 200px;

height: 100px;

border: 1px solid red;

/\*增加相对定位 position: relative\*/

position: relative;

top: 20px;

left: 20px;

}

#second {

width: 200px;

height: 100px;

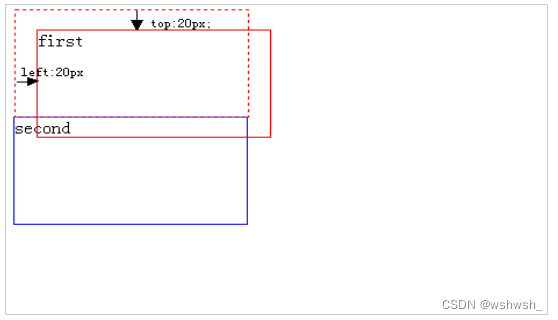
border: 1px solid blue;

}

</style>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png

偏移20px后：



虚线是初始的位置空间，现在看明白了吧，相对定位相对的是它原本在文档流中的位置而进行的偏移，而我们也知道relative定位也是遵循正常的文档流，它没有脱离文档流，但是它的top/left/right/bottom属性是生效的，可以说它是static到absoult的一个中间过渡属性，最重要的是它还占有文档空间，而且占据的文档空间不会随 top / right / left / bottom 等属性的偏移而发生变动，也就是说它后面的元素是依据虚线位置( top / left / right / bottom 等属性生效之前)进行的定位，这点一定要理解。

那好，我们知道了top / right / left / bottom 属性是不会对relative定位的元素所占据的文档空间产生偏移，那么margin / padding会让该文档空间产生偏移吗？答案是肯定的，我们一起来做个试验吧：

**3、添加margin属性：**

<style type="text/css">

#first {

width: 200px;

height: 100px;

border: 1px solid red;

position: relative;

top: 20px;

left: 20px;

/\* add margin\*/

margin: 20px;

}

#second {

width: 200px;

height: 100px;

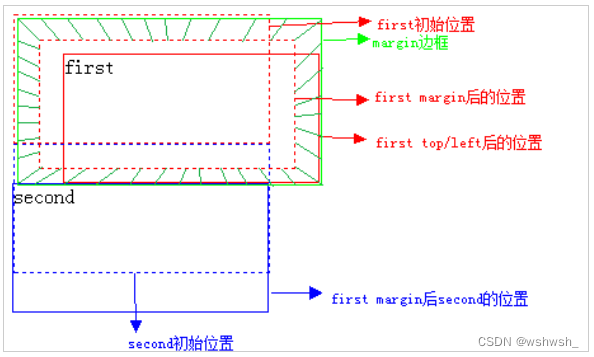
border: 1px solid blue;

}

</style>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png

设置margin：20px后：



对比一下，是不是就很清晰了，我们先将first元素外边距设为20px，那么second元素就得向下偏移40px，所以margin是占据文档空间！

**相对定位总结：**

* **相对定位相对的是它原本在文档流中的位置而进行的偏移**
* 元素不会脱离文档流，原来**在标准流的区域继续占有**，后面的盒子仍然以标准流的方式对待它

**4、绝对定位 absolute(重点)**

所谓绝对定位是元素在移动位置时，是相对于它祖先元素来说的。

**使用absoult定位的元素脱离文档流后**，就只能根据祖先类元素(父类以上)进行定位，而这个祖先类还必须是以postion非static方式定位的， 举个例子，a元素使用absoulte定位，它会从父类开始找起，寻找以position非static方式定位的祖先类元素(注意，一定要是直系祖先才算哦~），直到<html>标签为止，这里还需要注意的是，relative和static方式在最外层时是以<body>标签为定位原点的，而absoulte方式在无父级是position非static定位时是以<html>作为原点定位。<html>和<body>元素相差9px左右。

绝对定位一般和相对定位一起使用(子绝父相)

示例代码：

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>子绝父相</title>

<style>

div {

width: 100px;

height: 100px;

border: 1px solid black;

}

.nofather {

/\* 使用绝对定位 \*/

position: absolute;

top: 0;

left: 100px;

}

.father {

width: 200px;

height: 200px;

/\* 父元素使用相对定位 \*/

position: relative;

left: 100px;

}

.son {

/\* 子元素使用绝对定位 \*/

position: absolute;

left: 50px;

top: 30px;

}

</style>

</head>

<body>

<div class="nofather">我没有父元素</div>

<h1>内容</h1>

<h1>内容</h1>

<h1>内容</h1>

<h1>内容</h1>

<div class="father">

<div class="son">我有父元素</div>

</div>

<h1>内容</h1>

<h1>内容</h1>

<h1>内容</h1>

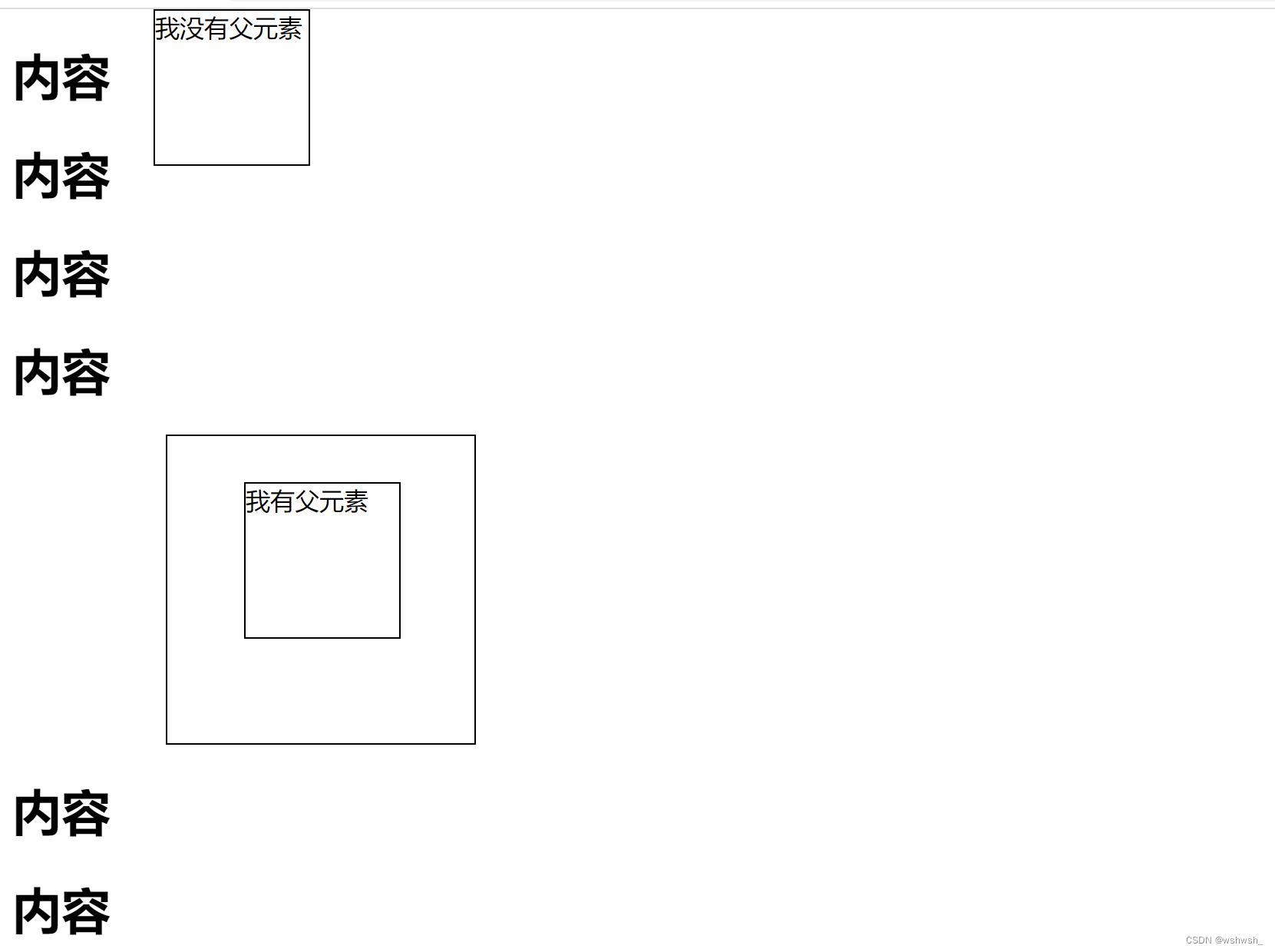
<h1>内容</h1>

<h1>内容</h1>

</body>

</html>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png



从运行结果上可以看出，使用绝对定位，如果没有父元素则是相对于body进行定位，如果有父元素，父元素需要使用相对定位，子元素使用绝对定位，子元素则会相对于父元素进行定位。并且使用绝对定位，元素会脱离文档流，后面的元素会自动进行填补。

**绝对定位总结：**

* **绝对定位不会保留原来的位置（脱离文档流），那后面盒子就会往上占了它的位置；**
* 如果**没有祖先元素**或者**祖先元素没有定位**，则**以浏览器为基准定位**；
* 如果**祖先元素有定位**（可以是相对、绝对或者固定定位），则以**最近**的**已经定位**的**祖先元素**为基准移动位置。
* **子绝父相** —— 虽然**父元素**的定位可以是相对、绝对或者固定定位，但是绝对定位和固定定位都会脱离标准流，所以我们实际开发中，**一般都使用相对定位**，这就是常说的**子绝父相**。

**5、固定定位 fixed(重点)**

固定定位相对于浏览器窗口，脱离文档流，使用fixed的元素不会随窗口的滚动而滚动

示例代码：

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>固定定位</title>

<style>

.fixed {

width: 200px;

height: 200px;

background-color: aqua;

/\* 使用固定定位 \*/

position: fixed;

top: 50px;

left: 300px;

}

</style>

</head>

<body>

<div class="fixed">

</div>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

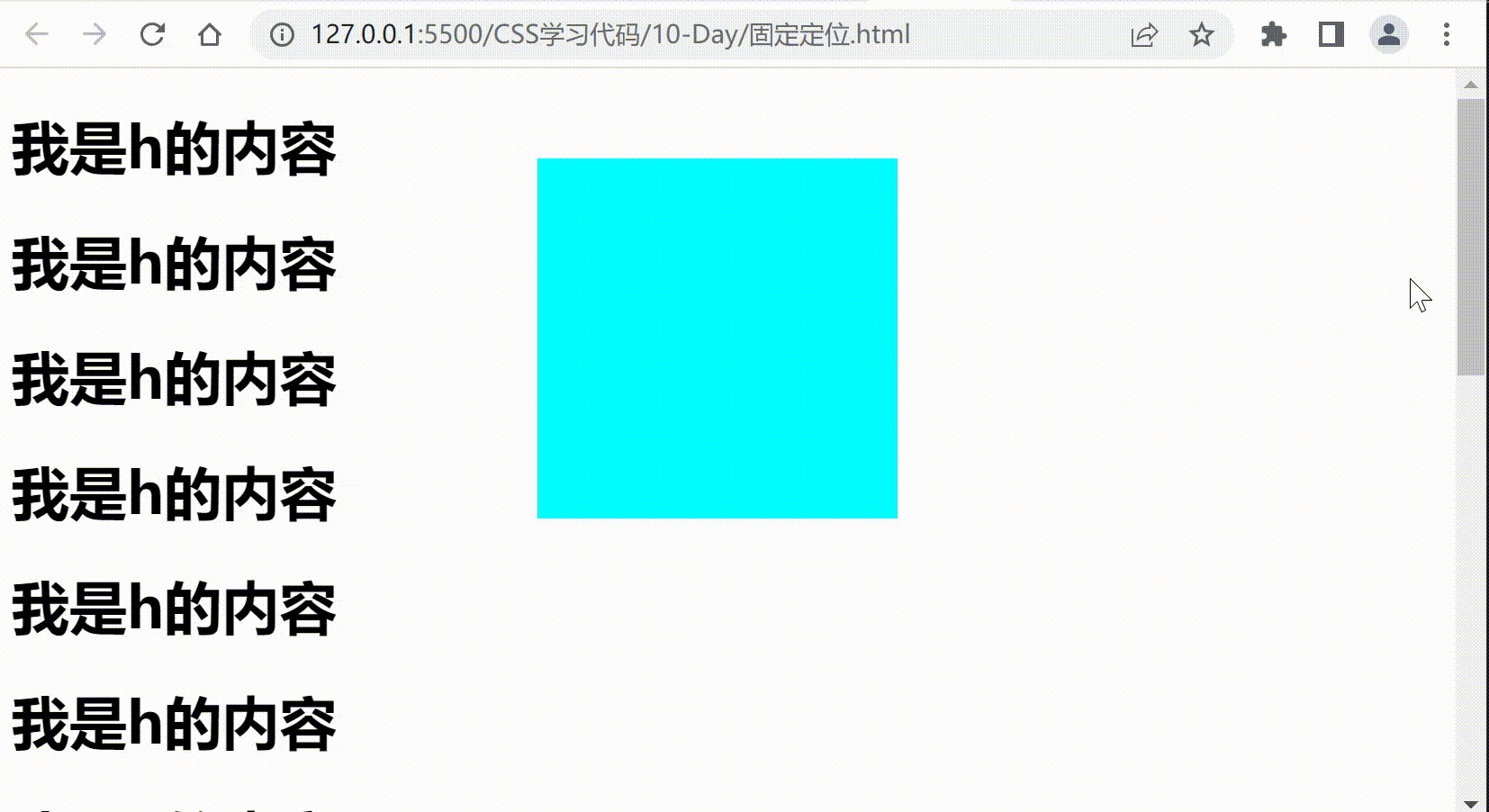
<h1>我是h的内容</h1>

<h1>我是h的内容</h1>

</body>

</html>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png



**固定定位总结：**

* **固定定位是参照浏览器窗口的左上角进行偏移。**
* **使用固定定位之后，元素会脱离标准文档流。**
* **使用固定定位之后，在滚动浏览器页面时，元素不会随着窗口的滚动而滚动。**

**6、粘性定位 sticky**

sticky 英文字面意思是粘，粘贴，所以可以把它称之为粘性定位。

position: sticky; 基于用户的滚动位置来定位。

粘性定位的元素是依赖于用户的滚动，在 **position:relative** 与 **position:fixed** 定位之间切换。

它的行为就像 **position:relative;** 而当页面滚动超出目标区域时，它的表现就像 **position:fixed;**，它会固定在目标位置。

元素定位表现为在跨越特定阈值前为相对定位，之后为固定定位。

这个特定阈值指的是 top, right, bottom 或 left 之一，换言之，**指定 top, right, bottom 或 left 四个阈值其中之一，才可使粘性定位生效。**否则其行为与相对定位相同。

**注意:** Internet Explorer, Edge 15 及更早 IE 版本不支持 sticky 定位。 Safari 需要使用 -webkit- prefix (查看以下实例)。

示例代码：

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>sticky 定位</title>

<style>

div.sticky {

position: -webkit-sticky;

position: sticky;

top: 0;

padding: 5px;

background-color: #cae8ca;

border: 2px solid #4CAF50;

}

</style>

</head>

<body>

<p>尝试滚动页面。</p>

<p>注意: IE/Edge 15 及更早 IE 版本不支持 sticky 属性。</p>

<div class="sticky">我是粘性定位!</div>

<div style="padding-bottom:2000px">

<p>滚动我</p>

<p>来回滚动我</p>

<p>滚动我</p>

<p>来回滚动我</p>

<p>滚动我</p>

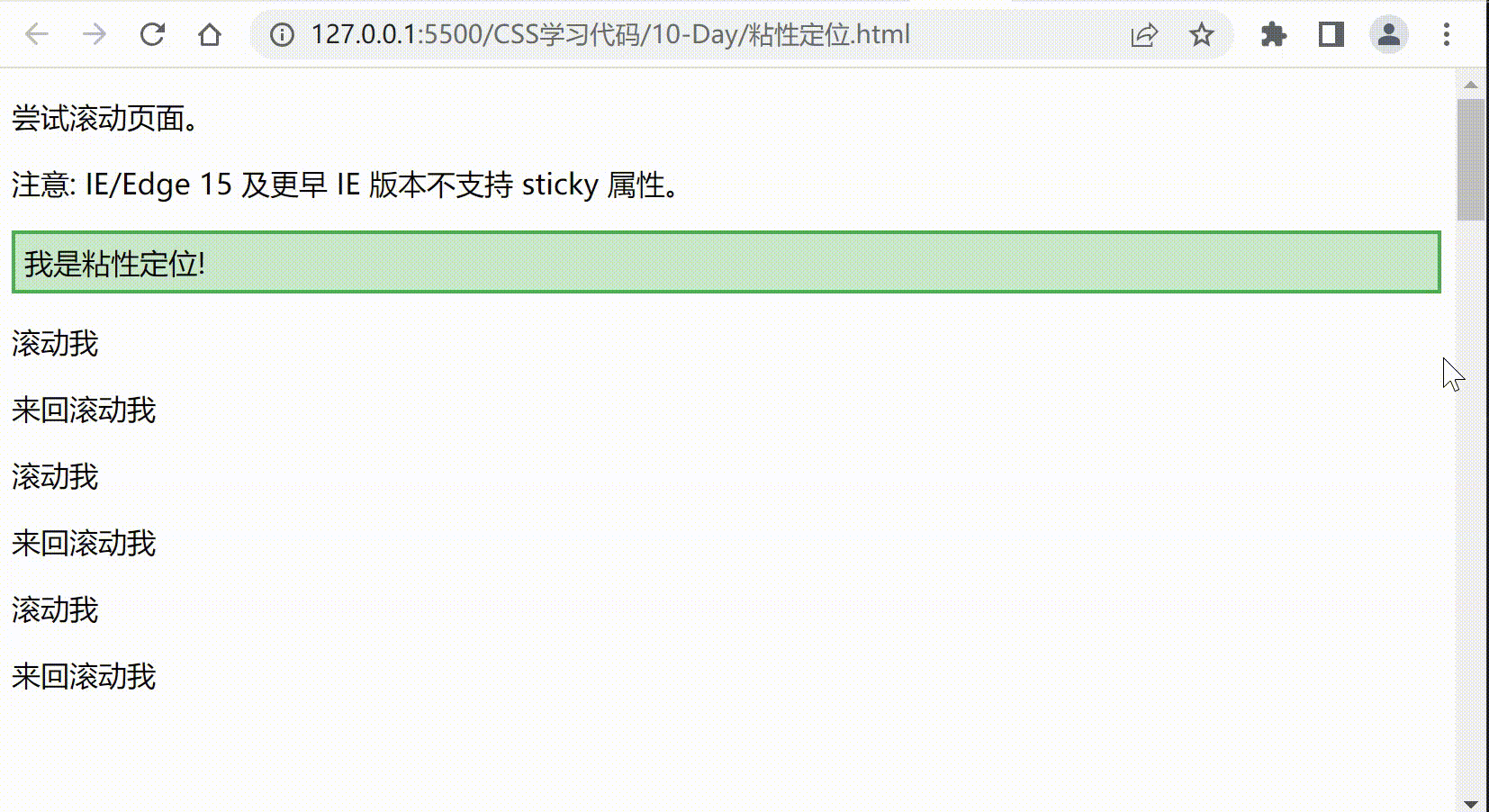
<p>来回滚动我</p>

</div>

</body>

</html>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png



**粘性定位总结：**

* **以浏览器的可视窗口为参照点移动元素（固定定位特点）**
* **粘性定位占有原先的位置（相对定位特点）**
* **必须添加 top 、left、right、bottom 其中一个才有效**

有时候设置会发现粘性定位不起作用，又找不到原因，可以看看有没有以下几点原因：

1. 父元素不能overflow:hidden或者overflow:auto属性。
2. 必须指定top、bottom、left、right4个值之一，否则只会处于相对定位
3. 父元素的高度不能低于sticky元素的高度
4. sticky元素仅在其父元素内生效

**7、重叠元素**

在对元素进行定位时，它们可以与其他元素重叠。

z-index 属性指定元素的堆栈顺序（哪个元素应放置在其他元素的前面或后面）。

元素可以设置正或负的堆叠顺序。

示例代码：

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>重叠元素</title>

<style>

.div1 {

width: 200px;

height: 200px;

background-color: red;

position: absolute;

top: 0;

/\* 值越大，元素就会显示在其他元素上面 \*/

z-index: 999;

}

.div2 {

width: 100px;

height: 100px;

background-color: yellowgreen;

position: absolute;

top: 150px;

left: 150px;

}

</style>

</head>

<body>

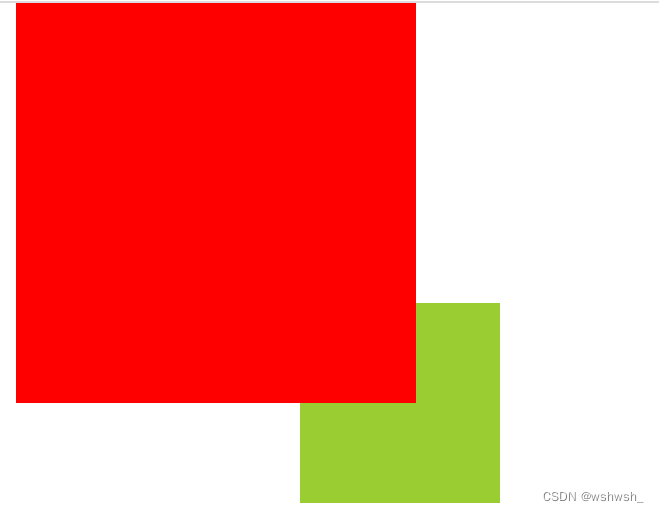
<div class="div1"></div>

<div class="div2"></div>

</body>

</html>

https://csdnimg.cn/release/blogv2/dist/pc/img/newCodeMoreWhite.png



注意：

* **z-index** **仅能在定位元素上生效(重点)**
* **如果两个元素为父子关系，则z-index也不生效**

**总结：**

position 属性规定应用于元素的定位方法的类型。

| **定位模式** | **是否脱离文档流** | **移动位置** | **是否常用** |
| --- | --- | --- | --- |
| static 静态定位 | 否 | 不能使用边偏移 | 很少 |
| relative 相对定位 | 否(占有原来位置) | 相对于自身位置偏移 | 常用 |
| absolute 绝对定位 | 是(不占有原来位置) | 带有定位的父级 | 常用 |
| fixed 固定定位 | 是(不占有原来位置) | 浏览器可视区 | 常用 |
| sticky 粘性定位 | 否(占有原来位置) | 浏览器可视区 | 偶尔用 |

1. 我们学习定位时，一定要记住 相对定位、固定定位、绝对定位 两个大的特点：

* 是否占有位置（是否脱离文档流）
* 以谁为基准进行移动位置。

1. 学习定位，重点要学会子绝父相，这是开发中最常用的。