包含块：在CSS中，元素都有一个框，需要设置border属性来显示框。框的定位和尺寸的计算都取决于一个矩形的边界，称为包含块。

“一个框的包含块”，指的是“该框所存在的那个包含块”，并不是它建造的包含块。

<div>

<table>

<tr>

<td>hi</td>

</tr>

</table>

</div>

DIV和TABLE都建造包含块。TABLE的包含块是指DIV建造的包含块，而TD的包含块指TABLE建造的包含块。

每个框关于它的包含块都有一个位置，但是它不会被包含块限制，它可以溢出。包含块上可以通过设置’overflow特性达到处理溢出’。

包含块的判断总体流程如下：



根元素：处于文档树最顶端的元素，没有父节点。在HTML中，根元素是html元素（有些浏览器是body元素）

初始包含块：根元素存在的包含块为初始包含块。

初始包含块的direction（CSS文本属性）属性与根元素相同。

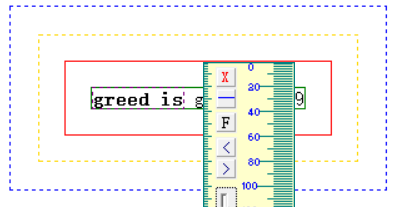
包含块判定及其范围：

http://w3help.org/zh-cn/kb/008.html

静态定位元素和相对定位元素

元素的定位属性’position’为’relative’(相对定位)或’static’(静态定位)，它的包含块由它最近的块级、单元格（table cell）或者行内框（inline-block）祖先元素的框创建。

<table id="table1" style="padding: 20px; border: 1px dashed blue;">  
 <tr style=" padding: 20px; border: 5px solid black;">  
 <td id="td1" style="padding: 20px; border: 1px dashed gold;">  
 <div id="div1" style="padding:20px;border:1px solid red;">  
 <span style="border: 1px solid green">  
 <strong id=”greed” style="position:relative; border: 1px dashed purple">greed is</strong>  
 good 999999  
 </span>  
 </div>  
 </td>  
 </tr>  
</table>



从图中可以看出<tr>的border属性没有作用。<td>的包含块为id=table1的<table>

从图中可以看出id=greed的<strong>元素的包含块为id=div1的<div>，<span>不是其包含块，原因是<span>为inline元素。

从图中可以看出<strong>的包含块的区域是div1的内容边界（content border），也就是内边界。这与后面的绝对定位是有区别的，绝对定位的包含块是祖先元素的内边距边界（padding border ）

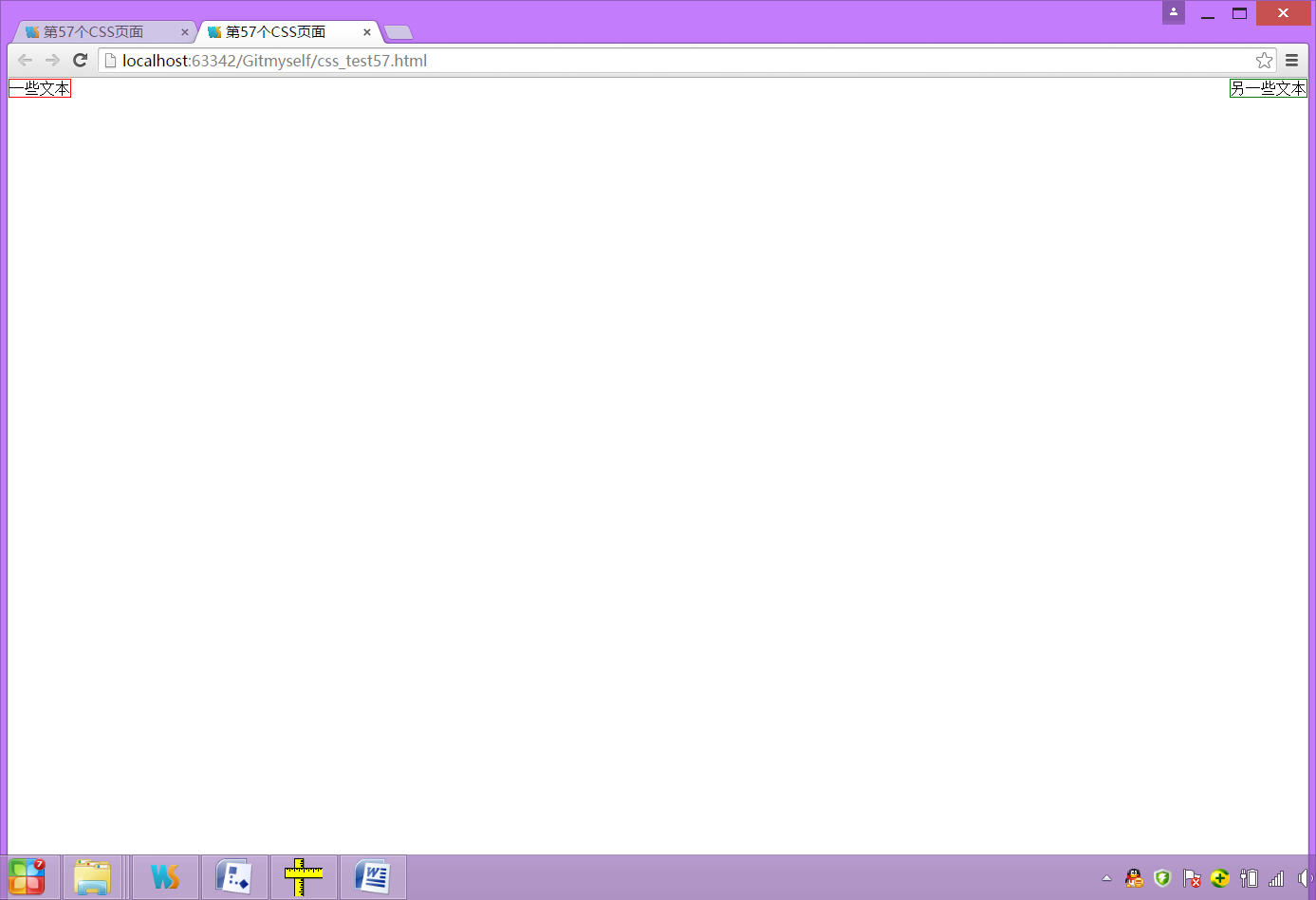
上述代码中，元素的包含块如下表所示：

|  |  |
| --- | --- |
| 元素 | 包含块 |
| table1 | body（body为table的祖先元素） |
| td1 | table1 |
| div1 | td1 |
| greed | div1 |

固定定位元素：

如果元素是固定定位（position:fixed）元素，它的包含块是当前可视窗口，即浏览器窗口本身。

<p style="position: fixed; left: 0px; top: -15px; border: 1px solid red; ">一些文本</p>  
<p style="position: fixed; right: 0px; top: -15px; border: 1px solid green;">另一些文本</p>



从图中可以看出，设置了left和right为0后，第1个<p>的左边位于视窗的左边，第2个<p>元素的右边位于视窗的右边。但是top值设置为-15px，<p>元素的顶部才位于是视窗的顶部。

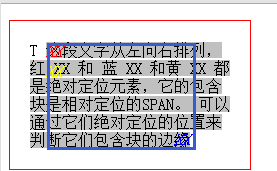
绝对定位元素：position:absolute

绝对定位元素的包含块由离它最近的position属性为absolute、fixed或relative的祖先元素创建。

如果符合条件的祖先元素为行内元素（<span>,<a>,<strong>,<img>,<textarea>,<input>），包含块取决于行内元素的direction属性：

1. 如果direction属性为ltr(文本从左向右)，

<p style="border: 1px solid red; width:200px;padding: 20px;">  
 T  
 <span style="background-color: #C0C0C0; position: relative;">  
 这段文字从左向右排列，红 XX 和 蓝 XX 和黄 XX 都是绝对定位元素，它的包含块是相对定位的SPAN。  
 可以通过它们绝对定位的位置来判断它们包含块的边缘  
 <em style="position: absolute; color: red; left: 0px; top: 0px">XX</em>  
 <em style="position: absolute; color: yellow; left: 0px; top: 20px">XX</em>  
 <em style="position: absolute; color: blue; right: 0px; bottom: 0px">XX</em>  
 </span>  
</p>

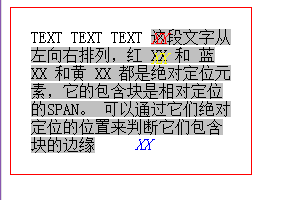


从图中可以看出，三个<em>的包含块都是<span>元素创建的，图中蓝色的方框就是包含块。按照标准来说，它们包含块的左顶边是<span>形成的第一个框（即第一行的灰色部分）的顶、左内边距边界，包含块的右、下边是<span>生成的最后一个框（最后一行灰色的部分）的右、下内边距边界。

包含块距离红色border的顶部距离为20px。

包含块的宽度为负：

<p style="border: 1px solid red; width:200px;padding: 20px;">  
 TEXT TEXT TEXT  
 <span style="background-color: #C0C0C0; position: relative;">  
 这段文字从左向右排列，红 XX 和 蓝 XX 和黄 XX 都是绝对定位元素，它的包含块是相对定位的SPAN。  
 可以通过它们绝对定位的位置来判断它们包含块的边缘  
 <em style="position: absolute; color: red; left: 0px; top: 0px">XX</em>  
 <em style="position: absolute; color: yellow; left: 0px; top: 20px">XX</em>  
 <em style="position: absolute; color: blue; right: 0px; bottom: 0px">XX</em>  
 </span>  
</p>



从图中可以看出绝对定位为bottom:0px,right:0px的蓝色的XX定位包含块的右下边界，绝对定位为left:0px,top:0px的红色的XX定位包含块的左上边界，可以看出包含块无法围城一个区域，宽度是负的。

<http://w3help.org/zh-cn/kb/008.html>

中描述的direction为rtl的情况在浏览器上测试的结果与ltr的情况相同。

祖先元素如果不是行内（display:inline）元素，包含块的元素应该是符合条件的祖先元素的内边距边界(padding border)

<div id="container" style="padding:50px; background-color:green; position:relative; width:200px; height:200px;">  
 <div id="div1" style="width:100%; height:100%; border:2px solid blue;">  
 <div id="content" style="border:1px solid red; position:absolute; left:0; top:0;">absolute element</div>  
 </div>  
</div>

虽然第2个<div>（蓝色border）是第3个<div>（红色border）的父元素，但其position为默认的static，所以其不创建第3个<div>的包含块。第3个<div>的包含块由第1个<div>(绿色背景)创建，为其内边距边界

