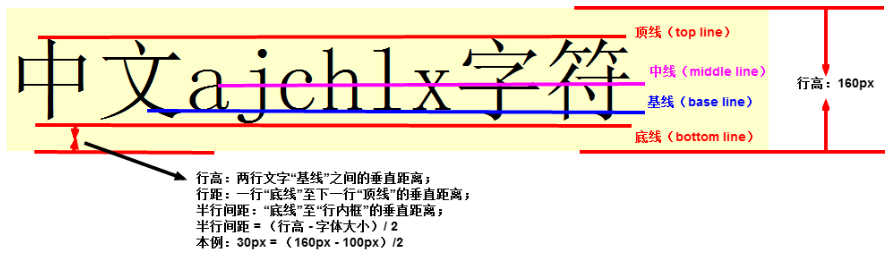
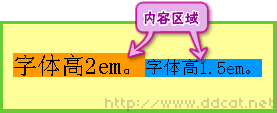
深入理解 CSS 中的行高与基线

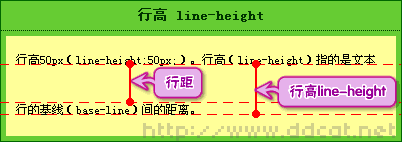
**1、基本概念**

**1.  基线、底线、顶线、中线**

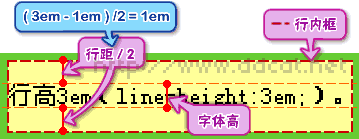
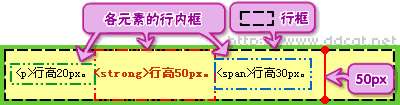


注意：基线（base line）并不是汉字文字的下端沿，而是英文字母“x”的下端沿。  
  
**2. 内容区**  


内容区是指底线和顶线包裹的区域（行内元素display：inline可以通过background-color属性显示出来），实际中不一定看得到，但确实存在。内容区的大小依据font-size的值和字数进行变化。

**3. 行距、行高**  


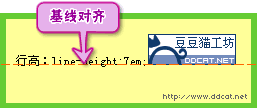
行高（line-height）：包括内容区与以内容区为基础对称拓展的空白区域，我们称之为行高。一般情况下，也可以认为是相邻文本行基线间的距离。

行距：指相邻文本行间上一个文本行基线和下一文本行顶线之间的距离。当然，我更愿意认为是（上文本行行高-内容区高度）/2+（下文本行行高-内容区高度）/2。  
  
**4. 行内框**  
  
行内框是一个浏览器渲染模型中的一个概念，无法显示出来，但是它又确实存在，它的高度就是行高指定的高度。  
  
**5. 行框**  


行框（line box），同行内框类似的概念，行框是指本行的一个虚拟的矩形框，也是浏览器渲染模式中的一个概念。行框高度等于本行内所有元素中行内框最大的值（以行高值最大的行内框为基准，其他行内框采用自己的对齐方式向基准对齐，最终计算行框的高度）。

**Vertical-align:垂直居中**

上一节我们讲解了行高与单行纯文字的垂直居中（line-height），而如果行内含有图片和文字，在浏览器渲染出来后，读者可以发现文字和图片在垂直方向**并不是沿中线居中，而是沿基线对齐**。这是因为，元素默认的垂直对齐方式为基线对齐（vertical-align: baseline）。



**CSS语法：vertical-align**

**语法：**

baseline | sub | super | top | text-top | middle | bottom | text-bottom | <百分比> | <长度> | inherit

**说明：**

设置元素内容的垂直对齐方式。

**参数：**

baseline： 基线对齐；

sub： 下标显示；

super： 上标显示；

top： 顶端对齐；

text-top： 文本的顶端对齐；

middle： 中部对齐； *//没有研究透的属性*

bottom： 底端对齐；

text-bottom：   文本的底端对齐；

百分比和长度： CSS2，可为负数。

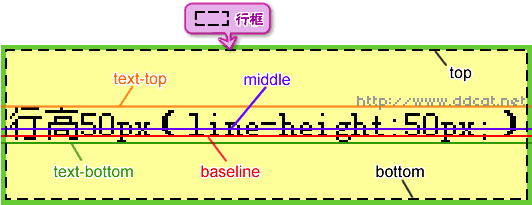
**初始值：**baseline

**继承性：**不继承

 此处需要特别注意的是：垂直对齐属性只对文本有效（是指包含了#Text节点的元素节点才能正确地处理vertical-align属性）。同时，该属性不能继承。

**属性值详解**

在上面一节中，介绍了文本的基线、顶线、中线和底线，还有内容区、行内框和行框，而本节的垂直对齐和这几个概念密切相关。



垂直对齐主要属性值的表现形式如上图所示。

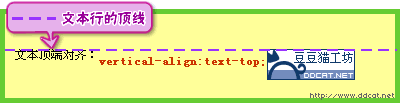
**1. 基线对齐（vertical-align : baseline）**

基线对齐（vertical-align : baseline）使元素的**基线**同基准元素（取行高最高的作为基准）的**基线**对齐  

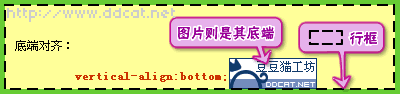

**2. 顶端对齐（vertical-align : top）**

顶端对齐（vertical-align : top）是将元素的行内框的**顶端**与行框的**顶端**对齐  


**3. 文本顶端对齐（vertical-align : text-top）**

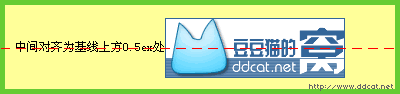
文本顶端对齐（vertical-align : text-top）是将元素行内框的**顶端**同文本行的**顶线**对齐  


**4. 底端对齐（vertical-align : bottom）**

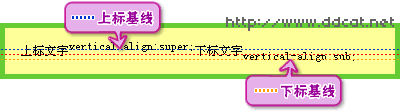
底端对齐（vertical-align : bottom）与顶端对齐（vertical-align : top）相反  


**5. 文本底端对齐（vertical-align : text-bottom）**  


**6. 中间对齐（vertical-align : middle）**

中间对齐（vertical-align : middle）通常使用在图片上，将图片的垂直方向的**中线**与文本行的**中线**对齐。（对于文字的处理有些偏差，具体依据还没有研究出来，有研究的同学可以联系我哦~~）  


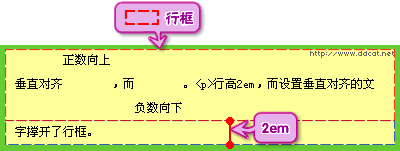
中线的定义为：中线位于基线的上方，**与基线的距离为小写字母x高度的一半（即0.5ex），而ex同font-size相关，**大部分浏览器认为1ex = 0.5em（em同样也是相对单位，不是绝对单位），因此会将基线以上四分之一em处作为中线来对齐。  
  
**7. 上标和下标**

上标（vertical-align:super）使元素的**基线**相对于基准元素的**基线**升高，下标（vertical-align:sub）使元素的基线降低，移动的幅度CSS规范中没有规定，由浏览器来决定。  


上下标不会改变元素文字的尺寸大小。  
  
**8. 长度值和百分比**

和上下标类似，长度值和百分比值可使元素的**基线**相对于基准元素的**基线**升高（正值）或者降低（负值）。

上下标的移动尺寸是由浏览器确定的，而设定长度值或者百分比，可以精确控制文字上下移动的幅度。

**百分比与行高有关（line-height）**，例如有如下代码，其显示如图7-44所示。  
  


**我的测试  @FireFox下**

**测试代码：**

<style type="text/css">  
 p {vertical-align:baseline; font-size:20px; line-height:60px; background-color:yellow;}  
 span {background-color: red;}  
 u {background-color: blue;}  
 del {background-color: pink;}  
</style>

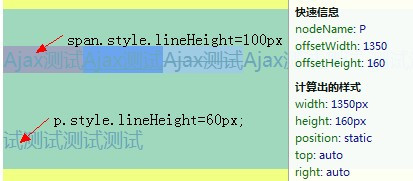
//HTML代码  
<p><span>Ajax测试</span><u>Ajax测试</u><del>Ajax测试</del>Ajax测试测试测试测试测试测试测试测试测试测试测试测试测试测试测试</p>

**默认情况：**  
  
   
**其他说明：**  
**1、**SPAN、U、DEL标签的 **offsetWidth = SUM（ 字符 \* font-size \* 修正系数）**（这里，中文的修正系数为1，数字的修正系数0.6，英文字符修正系数差异很大，比如ijl很小，wmk等比较大，大写英文同样修正系数不统一）。

**2、**SPAN、U、DEL标签的 **offsetHeight**。  
   推论：inline元素的背景渲染区域即内容区域大小，直接受到font-size的影响。  
         对<p>块级元素，块级元素的计算值高度由包含的行框高度累加而成，因此这里height为60px；

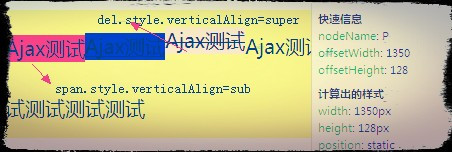
**3、**把 **span.style.lineHeight 设置为 15px(从10px变化到60px) ----> 发现没有变化**  
   推断：内容区大小不受line-height影响，line-height用于处理相邻文本行基线间的距离。

**4、**把 **span.style.lineHeight 设置为 70px(从61px变化到80px) ---->行框高度开始随着设置调整**  
   推断：行框高度是行内最高的行内框高度，通过line-height调整。  
         p元素的height计算值为 span.style.lineHeight值，不由p.style.lineHeight控制。  
   推断二：不设定height属性的<p>的height计算值为行框高度累加值。



场景4：line-height功能测试

**5、**把 **span.style.verticalAlign= sub；del.style.verticalAlign= super；-->看图说话**  
   推断：增加了行框的高度，上下标基于基线进行移动。



 场景5：sub、super垂直居中效果的研究

**6、**所有对齐方式的确认：  
    1、首先确认该行内的基准元素，取line-height值为最大值的元素作为基准；  
    2、其他文本向基准元素对齐，依据line-height和vertical-align来实现效果；  
    3、sub、super是改变基线的方式，因此会对行框的最终高度产生影响；  
    4、top、bottom是行内框对齐，top指该元素行内框的顶端与基准行内框顶端对齐；  
    5、text-top、text-bottom同样会影响最终行框的高度，是指该元素行内框的顶端与基准元素的内容区顶端对齐（当line-height=内容区高度时，跟基准内容区域顶端对齐。line-height小于内容区域高度时，会出现文本继续上移的现象，line-height设置为0px时，正好是内容区域的垂直中部跟基准内容区域顶端对齐。）  
  
    6、百分数和长度值：基于基线进行移动，百分数的计算值为行高\*百分数。

    7、关于middle，感觉是根据一条基准线进行对称拓展，但基准线产生的规则不清楚。