

CSS排版原理：行内元素、块级元素 与line-height

窦连军 @八月虎baidu

北京乐美无限科技有限公司

行内元素 和 块级元素

什么是块级元素和行内元素？

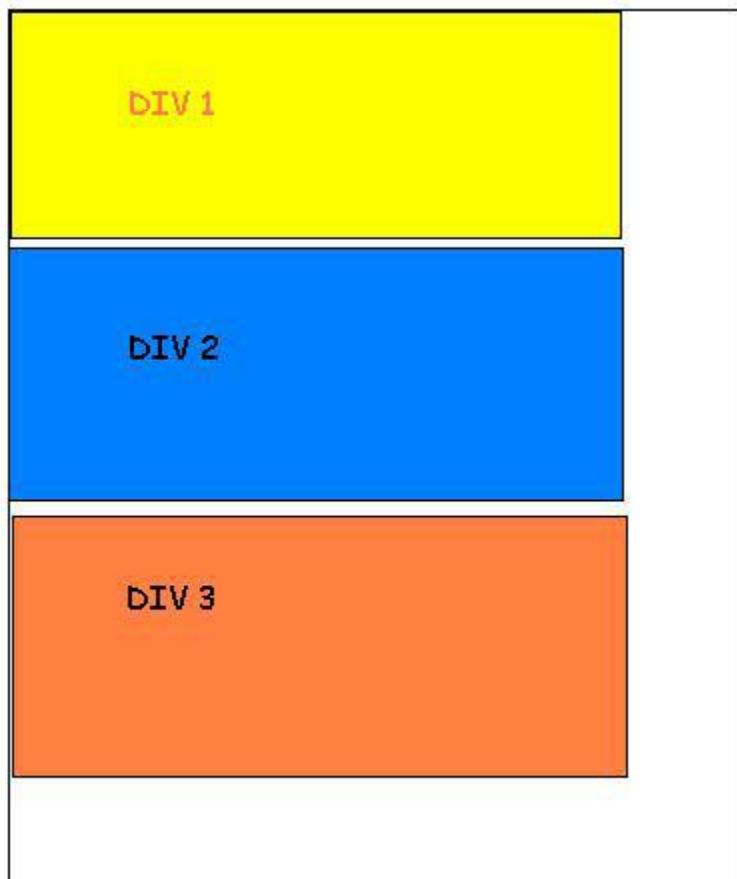
- block元素就是以“块”表现的元素 (block-like elements)

*- p、h1~h6、ul、ol、pre、dl、div、noscript、
blockquote、form、hr、table、fieldset、address*

- inline元素可以包含：文本、行内tag或其他 inline-block元素

块级元素

- 块级元素, 包括 DIV, P, H#, OL/UL等, 浏览器显示时, 一个叠加在另一个之上.



HTML

```
<body>  
  <div id="div1"></div>  
  <div id="div2"></div>  
  <div id="div3"></div>  
</body>
```

CSS

```
#div1 { width:300px;background:yellow;}  
#div2 { width:300px;background:blue;}  
#div3 { width:300px;background:orange;}
```

块级元素的特点

- 前后自动加换行符，总是独占一行；
- 宽度（width）缺省是它所在父元素内容区的100%，即：总是横向占满一行，除非设定一个宽度值。
- 高度（height）缺省由浏览器根据block自身的内容来计算出实际高度，即：其实际内容所占的高度。
- 高度（height），行高（line-height）以及顶、底内外边距都可控制；

什么是行内元素？

- inline元素即是以“行内”表现的字符级标签元素与文本内容（character level elements and text strings）
 - *#pcdata*（即文本）、*tt*、*i*、*b*、*big*、*small*、*em*、*strong*、*dfn*、*code*、*samp*、*kbd*、*var*、*cite*、*abbr*、*acronym*、*a*、*img*、*object*、*br*、*script*、*map*、*q*、*sub*、*sup*、*span*、*bdo*、*input*、*select*、*textarea*、*label*、*button*

行内元素

- 行内元素 包括span, img, input, a等, 一个接一个横向排列在一行上 在浏览器上显示, 只有当浏览器（父盒子内容区）的宽度不够时, 才自动换行.

This is small text and **this is big** *I am Italic*

```
<div id="row1" >
  <span class="norm">This is
small text and </span>
  <span class="big">this is
big</span>
  <span class="italicText"> I am
Italic</span>
</div>
```

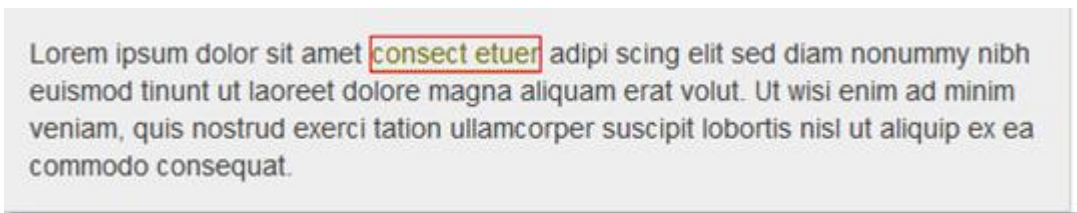
```
.norm {
    color:red;
}
.big {
    color:blue;
    font-weight:bold;
}
.italicText {
    color:green;
    font-style:italic;
}
#row1 {
    padding:10px;
    border:solid 1px #000;
}
```

行内元素的一般特点

- 后面不加换行符；
- 设置宽度（width）无效，实际宽度就是它所容纳的文字或图片的宽度。
- 设置高度（height）无效，**但可以通过line-height来设置其在行内的所占高度。**
- 设置上下内边距无效，设置上下外边距无效。即使设置了上下内边距和上下外边距，对元素周围的内容也没有影响的。
- 左右内边距可影响元素的**总宽度**，内边距可影响元素的边框；左右外边距可影响元素之间的距离。
- 行内元素可以作为任何其他元素的子元素。

行内元素

- 对<a>标签的height和width属性的修改:



- 设置<a>标签的属性padding:50px，对左右的内容有影响，但是对上下没影响。



特殊的行内元素：可置换元素/可变元素

- 可置换元素（Replaced element）：他们区别一般inline元素（相对而言，其他元素称non-replaced element）是：这些元素拥有内在尺寸（intrinsic dimensions），他们可以设置width/height属性。他们的性质同设置了display:inline-block的元素一致。
- **可置换元素**主要是指 等这类默认就img, input, textarea, select, object, button有 CSS 格式化外表范围的元素。虽然是行内元素，但margin、padding依旧可以影响到他的上下左右！**具有完整的盒模型特征。**

重要属性：

display

摆脱元素类别限制

display 属性

none

block

run-in

marker

inline-table

table-row-group

table-footer-group

table-column-group

table-cell

inline

list-item

compact

table

inline-block

table-header-group

table-row

table-column

table-caption

display:none 与 visibility:hidden

visible : 显示(缺省值).

hidden : 不显示 (但仍然占据屏幕区域)

This is small text and **this is big** *I am Italic*

```
.big {  
    visibility:hidden;  
}
```

This is small text and *I am Italic*

与 **display : none**的区别！

用display属性摆脱元素类别限制

- 通过CSS的display属性，我们可以摆脱上面表格里HTML标签归类的限制，自由地在不同标签/元素上应用我们需要的属性。
- 例如，我们可以对块元素[ul]标签加上 display:inline 属性，让原本垂直的列表水平显示，这在我们设置Blog导航条时得到了广泛应用；我们也完全可以把行内元素[cite]加上 display:block 这样的属性，让它也有每次都从新行开始的属性。

将块元素转变为行内元素

```
<html>
<head>
  <style type="text/css">
    p {
      display: inline
    }

    div {
      display: none
    }
  </style>
</head>
```

```
<body>
<p>本例中的样式表把段落元素设置为行内元素。</p>
```

```
<p>而 div 元素不会显示出来！</p>
```

```
<div>div 元素的内容不会显示出来！</div>
</body>
</html>
```

查看结果：

本例中的样式表把段落元素设置为内联元素。而 div 元素不会显示出来！

将行内元素转变为块元素

```
<html>
<head>
  <style type="text/css">
    span
    {
      display: block
    }
  </style>
</head>
<body>
```

查看结果:

本例中的样式表把 span 元素设置为块级元素。
两个 span 元素之间产生了一个换行行为。

本例中的样式表把 span 元素设置为块级元素。

两个 span 元素之间产生了一个换行行为。

```
</body>
</html>
```


display: inline-block

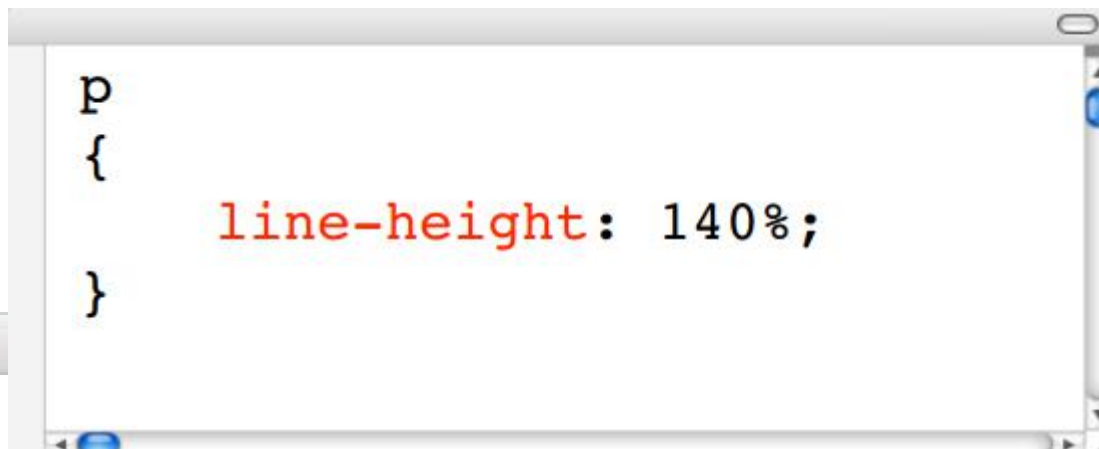
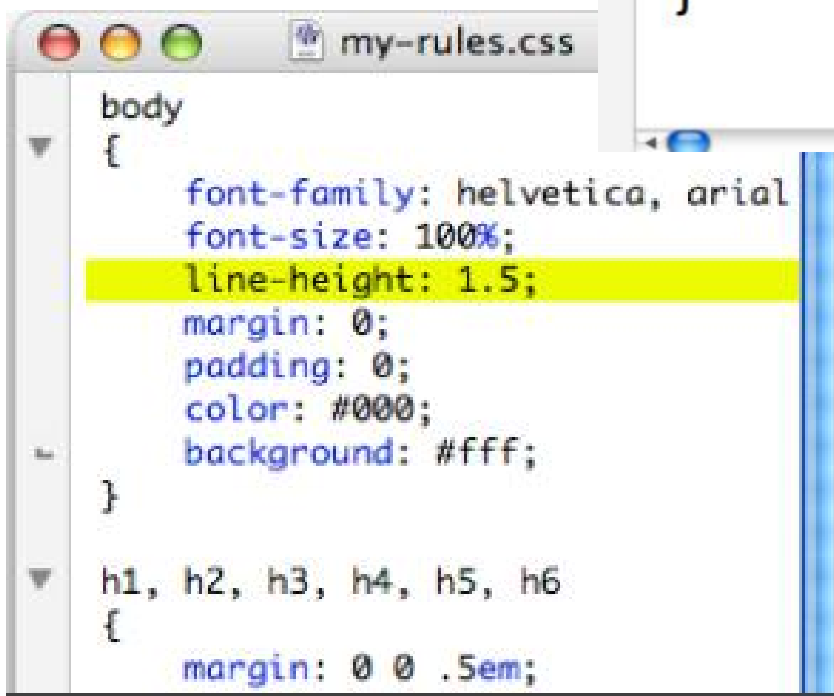
- 通过设置display属性为“inline-block”,使该元素保持为行内元素的特征,同时在内容区表现为块级容器。换句话说,inline-block 元素外部具有 inline 元素默认不换行的特性,内部的内容区同时又具有 block 元素可以设置宽高的特性。一般行内元素是不能设置宽和高。
- inline-block 元素还可以设置 vertical-align 属性,该属性设置自身盒子于所在行中与其他元素的对齐方式。

重要属性：

`line-height`

Line-height 行间距属性

- 浏览器缺省值为1.0-1.2倍字体大小。可如下自定义



line-height属性值没有单位时，表示字体大小的倍数。

深入理解：

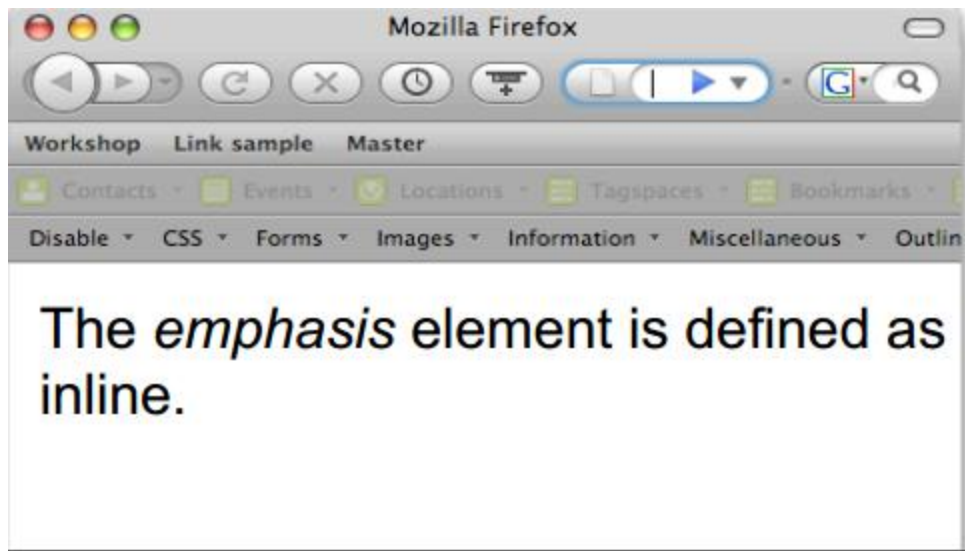
line-height

一个例子，看4种CSS盒子

<p>

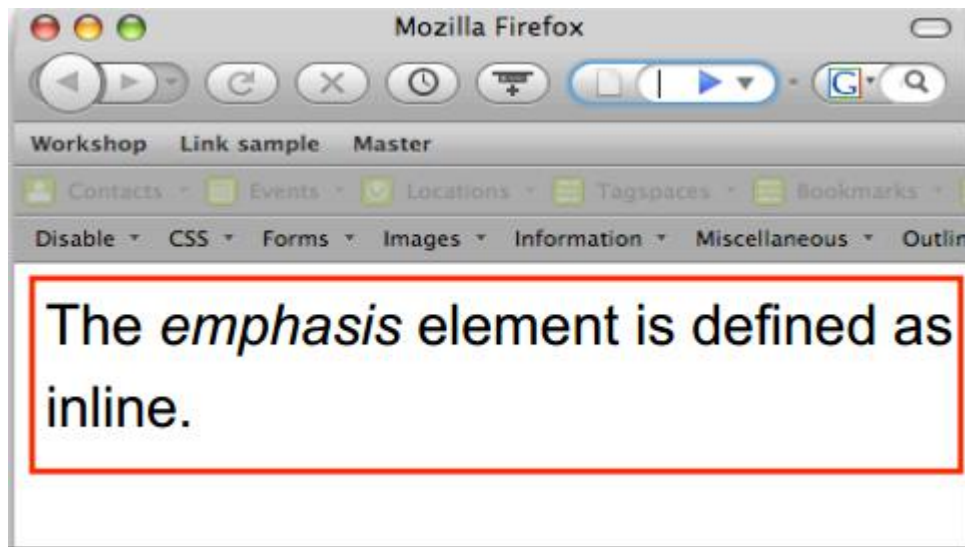
The emphasis
element is defined as
"inline".

</p>



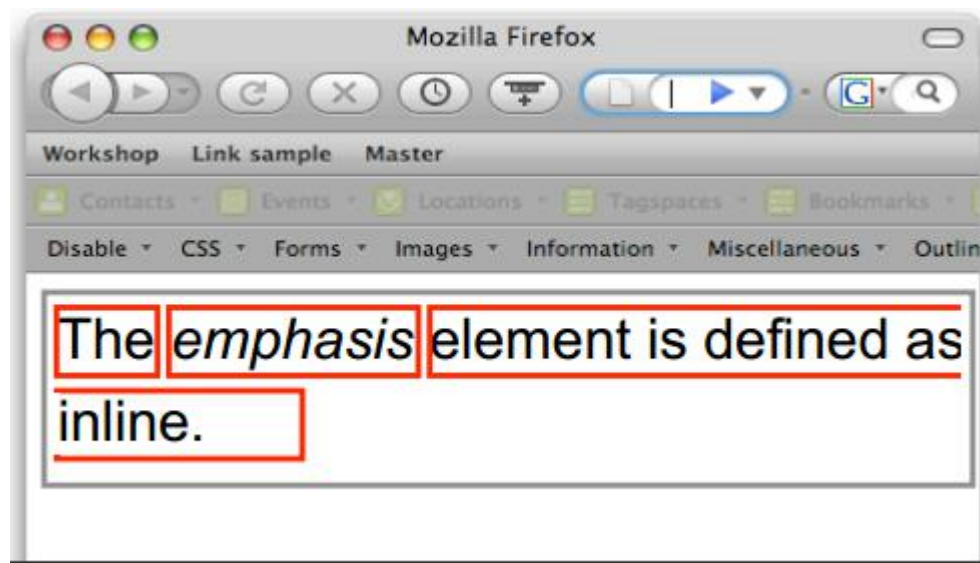
第一种：容器盒子

- P元素包含了其他元素，是其它元素的容器，因此称为“容器盒子”。



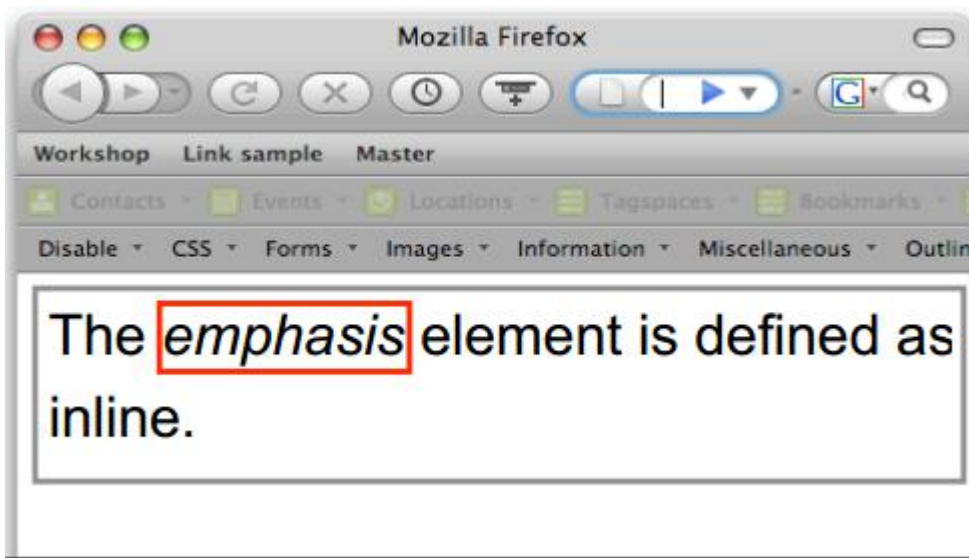
第二种：行内盒子

- 在P内部，是一组行内的元素盒子，称为行内盒子
- 行内盒子在自己的内容区不再生成新的盒子。浏览器将各个行内盒子的内容依次渲染在行上。



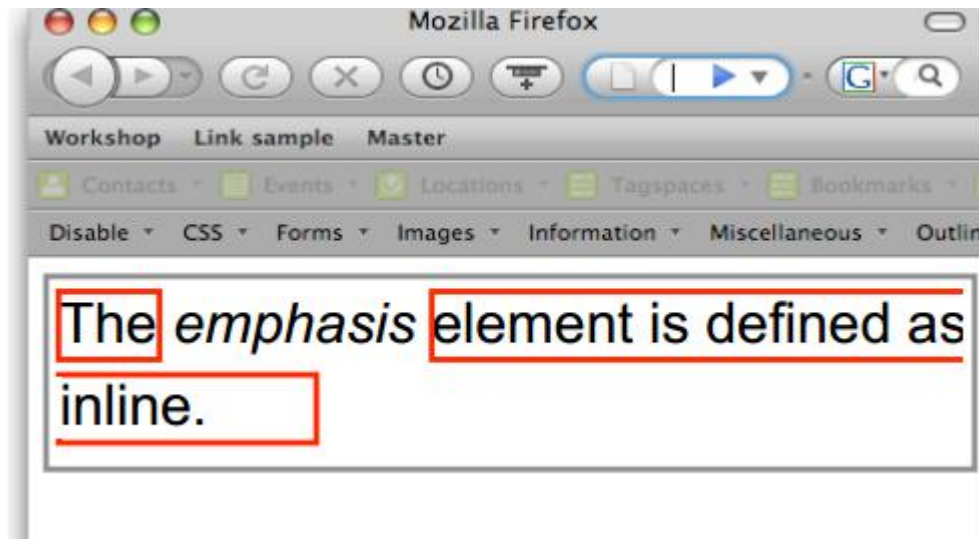
第二种：行内盒子（续）

- em元素就是一种“行内盒子”



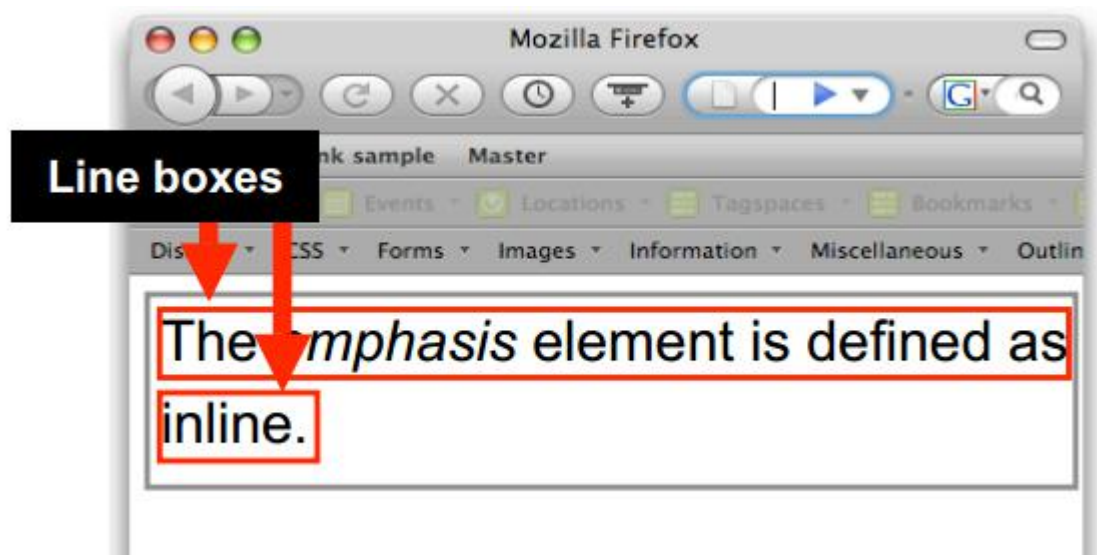
第二种：行内盒子（续）——普通文本

- 没有用标签引用起来的普通文本，被作为一种“匿名行内盒子”称呼。



第三种：整行盒子

- 在容器盒子中，各个行内盒子一个挨一个依次排成一行或多行，这就形成了“整行盒子”。



第四种：内容区盒子

- 内容区盒子是不可见的盒子，其环绕着文本。它的高度由字体的高度决定。



行内盒子
和 line-height

行高（line-height）与行内盒子

- 一个简单的计算公式：



步骤1

- 通过字体大小和line-height比较，来决定行间距
- 例如：

Font-size: 16px
Line-height: 20px
Difference: 4px (leading)

步骤2:

- 计算行距的一半值

$$4\text{px leading} / 2 = 2\text{px half-leading}$$

步骤3：将一半的行间距叠加在文字内容区的上边和下边。



但有时事情会变得有点复杂...



行内盒子（**inline box**）通常包裹着
文本内容区盒子， 半行距位于内容区
盒子的上边和下边。



行内盒子

然而, 行内盒子如果比内容区小了,
会怎么样?



例如，如果line-height 比font-size要小，
这种情况下，行内盒子将依照line-height

例如：

Font-size: 16px

Line-height: 12px

Inline box size: 12px

内容区将戳穿行内盒子的顶部和底部。
半行距将折叠在一起，构成行内盒子
的高度。



若干注意事项

整行盒子的高度计算

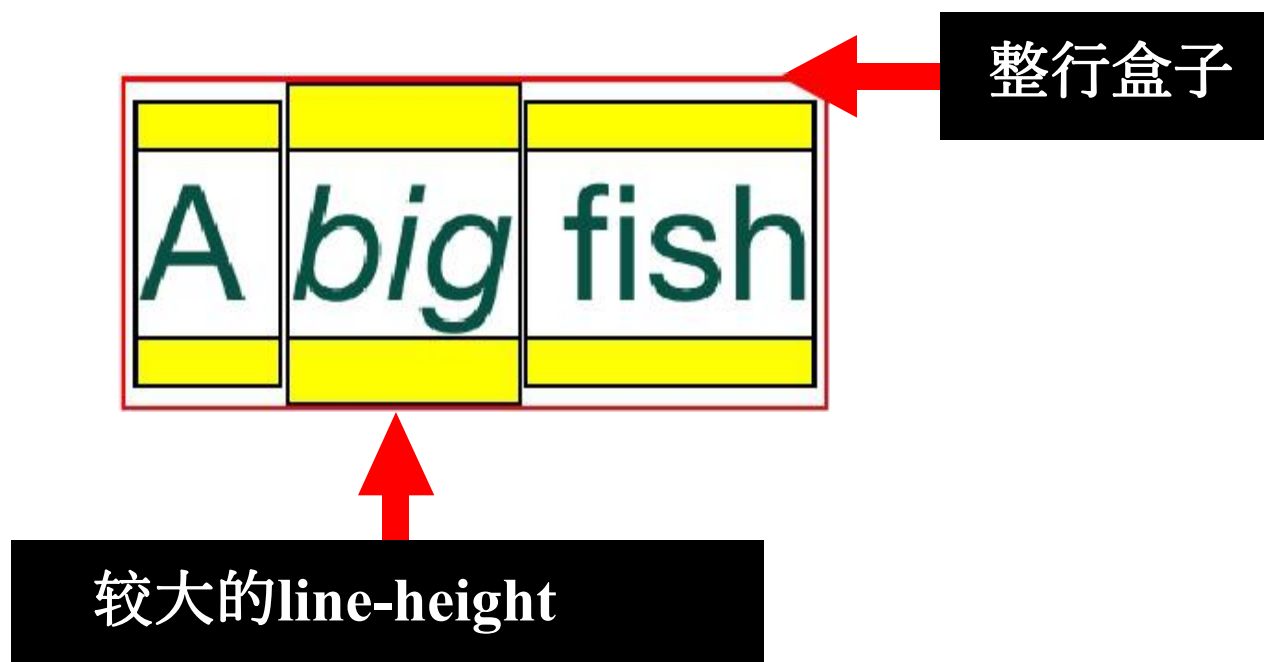
- 整行的高度由行内盒子（或者可替换元素）的最高值决定。



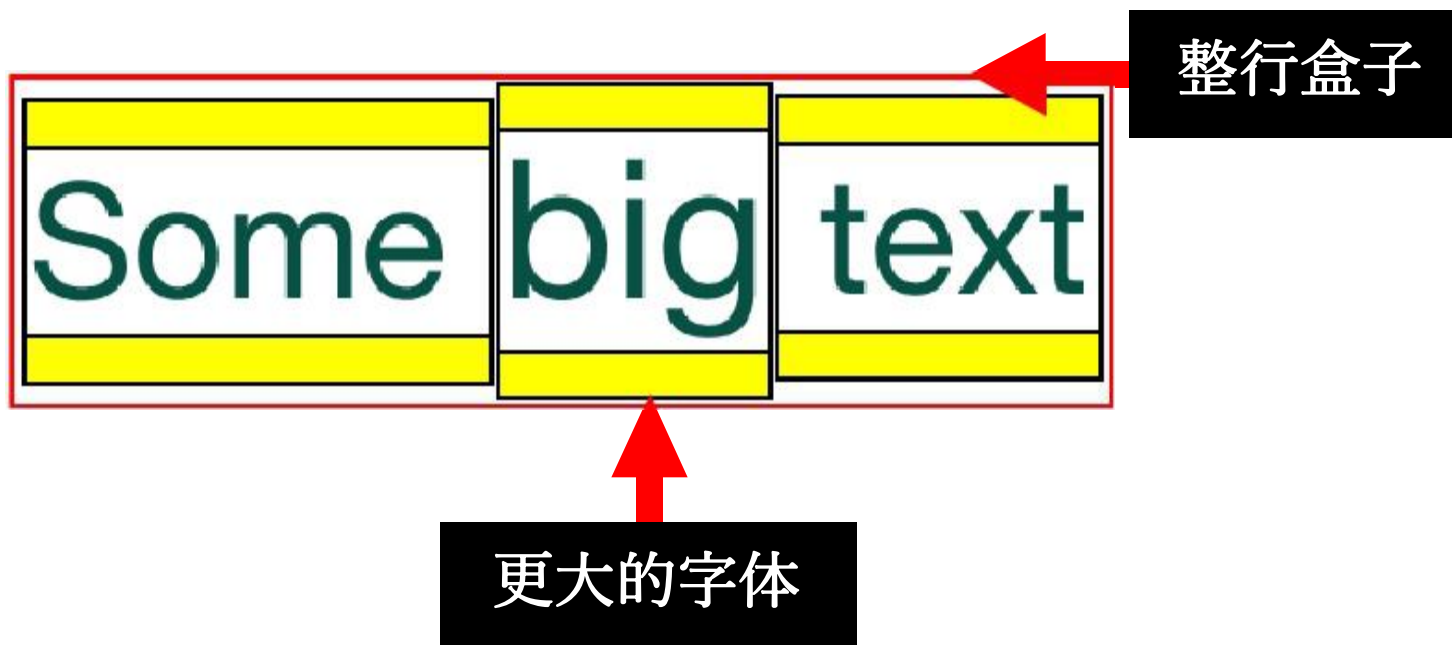
最高的行内盒子可能是：
匿名行内盒子



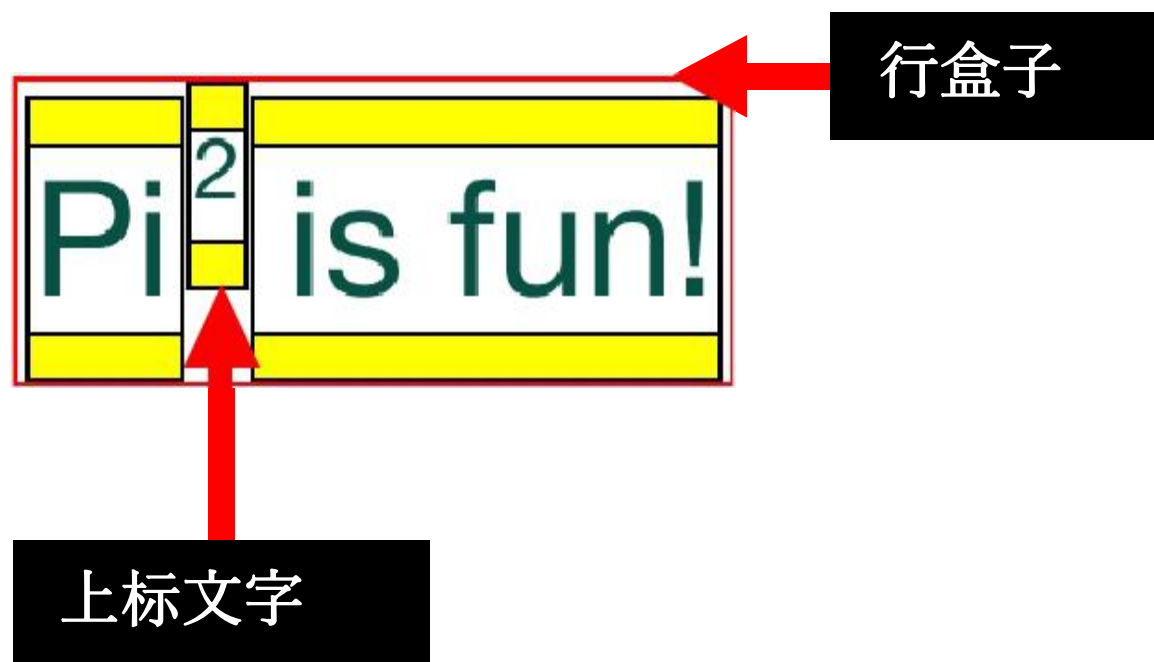
也可能是较大的**line-height**构成的行内盒子：



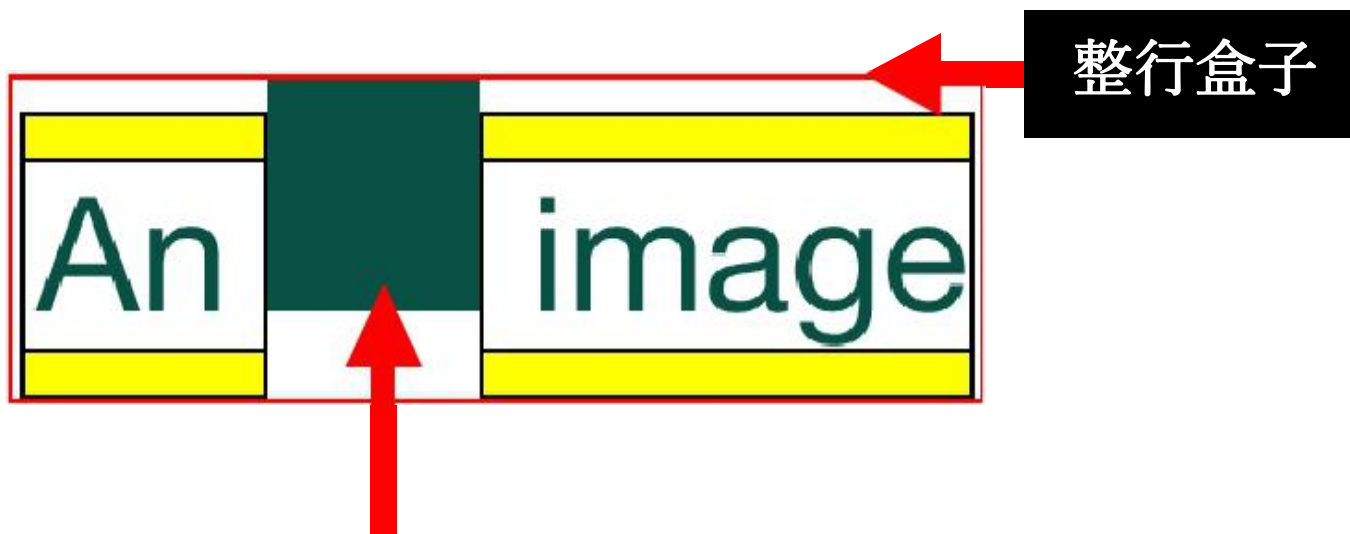
或某个行内盒子，其具有：
更大的font-size



或者某个上标文字、下标文字



或者是某个可替换元素， 比如：img图片

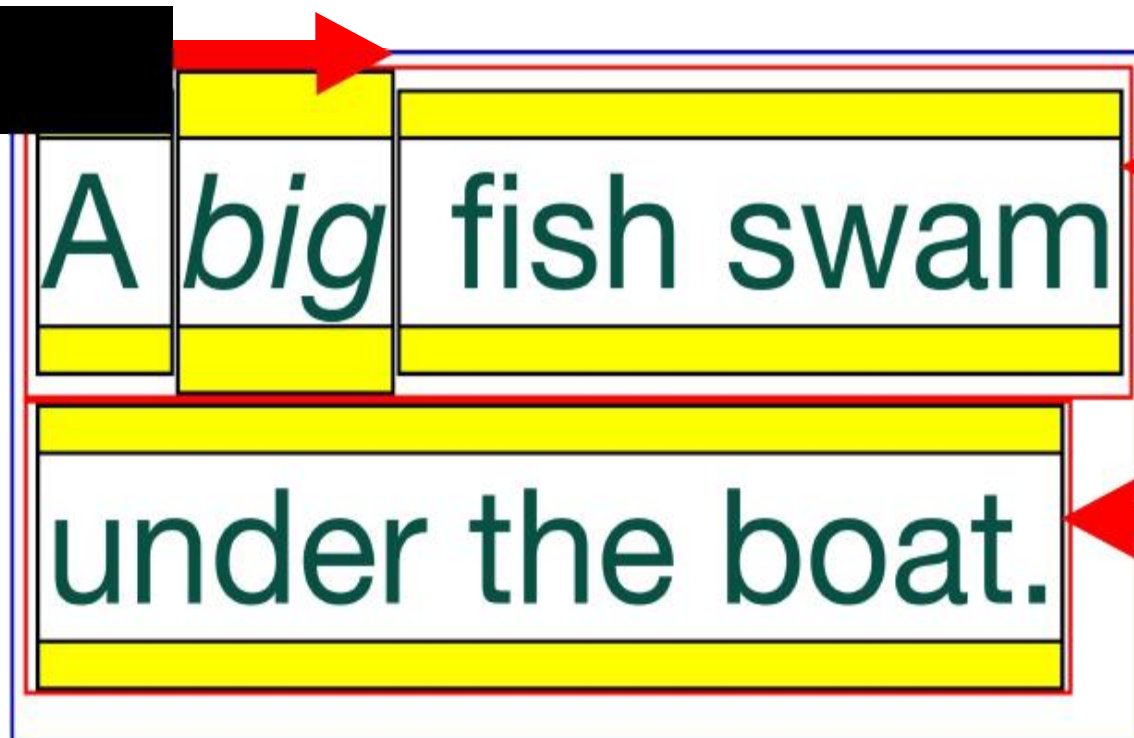


行内的image图片与基线（baseline）对齐

整行盒子在容器盒子的宽度
许可范围内，一个接一个依次垒加：

.

容器盒子



行盒子

行盒子

某些说明：关于 上标和下标

上标 和 下标元素

某些时候可能会令整行盒子显得太高。



你可以通过设置这些元素的**line-height**为“0”来修正它，这 will 把半行距从这些元素上移走。



```
sup, sub  
{ line-height: 0; }
```


Line-height在font中快捷用法



重要属性：

vertical-align

vertical-align用法

Formal syntax: baseline | sub | super | text-top | text-bottom | middle | top | bottom | <percentage> | <length>

```
vertical-align: baseline      /* keyword values */
vertical-align: sub
vertical-align: super
vertical-align: text-top
vertical-align: text-bottom
vertical-align: middle
vertical-align: top
vertical-align: bottom
vertical-align: 10em         /* <length> values */
vertical-align: 4px
vertical-align: 20%          /* <percentage> values */

vertical-align: inherit
```

vertical-align只可应用于行内元素和**table-cell**元素自身，不可用于块元素。

vertical-align: baseline

```
img { vertical-align: baseline; }
```

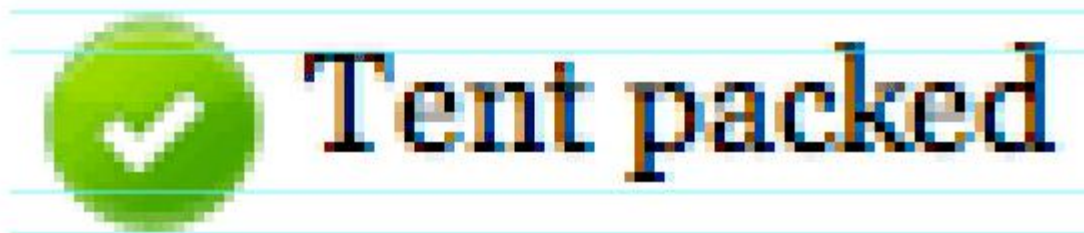
baseline为vertical-align属性的缺省值。图片会与文字在文本的baseline线上对齐。



vertical-align: middle

```
img { vertical-align: middle; }
```


- ✓ Tent packed
- ✓ Sleeping bag rolled
- ✓ Tickets secured



middle为vertical-align属性的最常见的用法。一般用于icon小图标与文字的对齐。浏览器会负责将图标与文本进行垂直居中对齐。

vertical-align: middle

```
img { vertical-align: middle; }
```

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante.  Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.. Vestibulum erat wisi, condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentum, elit eget tincidunt condimentum, eros ipsum rutrum orci, sagittis tempus lacus enim ac dui. Donec

有时图片高度超过了line-height或字体高度，则会拉大行高。

vertical-align: text-bottom

vertical-align: text-top



Tent packed

TEXT-BOTTOM



Tent packed

TEXT-TOP

`vertical-align: bottom`
`vertical-align: top`

对齐到整个行的顶部和底部。行的顶部和底部由行内最高行内元素的高度决定。

vertical-align: sub
vertical-align: super

Pellentesque habitant **VERTICAL ALIGN: SUPER;**

Pellentesque habitant **VERTICAL ALIGN: SUB;**

对齐到整个行的顶部和底部。行的顶部和底部由行内最高行内元素的高度决定。

vertical-align: <length>

将行内元素的**baseline**置于父元素的**baseline**之上的若干长度距离。

vertical-align: <百分比>

将行内元素的**baseline**置于行高的百分比。

table-cell布局中的vertical-align

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.	Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.
---	---

单元格的vertical-align的默认值是middle。

table-cell布局中的vertical-align

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.	Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.
Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.	Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

单元格的vertical-align的可以使用bottom和top来修饰，其他值则没有意义，并可能在各浏览器中表现不一致。

<td valign=top> 有了CSS之后，这种用法就已经过时了。

inline-block元素中的vertical-align

inline-block元素与img元素表现类似。可以设置其height和width。

结束了!