**Flex属性**

## 语法：

**flex**：none | <' [flex-grow](http://css.doyoe.com/properties/flex/flex-grow.htm) '> <' [flex-shrink](http://css.doyoe.com/properties/flex/flex-shrink.htm) >'? || <' [flex-basis](http://css.doyoe.com/properties/flex/flex-basis.htm) '>

**默认值**：看各分拆属性

**适用于**：flex子项

**继承性**：无

**动画性**：否

**计算值**：看各分拆属性

## 取值：

none：

none关键字的计算值为: 0 0 auto

<' [flex-grow](http://css.doyoe.com/properties/flex/flex-grow.htm) '>：

用来指定扩展比率，即剩余空间是正值时此「flex子项」相对于「flex容器」里其他「flex子项」能分配到空间比例。  
在「flex」属性中该值如果被省略则默认为「1」

<' [flex-shrink](http://css.doyoe.com/properties/flex/flex-shrink.htm) '>：

用来指定收缩比率，即剩余空间是负值时此「flex子项」相对于「flex容器」里其他「flex子项」能收缩的空间比例。  
在收缩的时候收缩比率会以伸缩基准值加权  
在「flex」属性中该值如果被省略则默认为「1」

<' [flex-basis](http://css.doyoe.com/properties/flex/flex-basis.htm) '>：

用来指定伸缩基准值，即在根据伸缩比率计算出剩余空间的分布之前，「flex子项」长度的起始数值。  
在「flex」属性中该值如果被省略则默认为「0%」  
在「flex」属性中该值如果被指定为「auto」，则伸缩基准值的计算值是自身的 <' [width](http://css.doyoe.com/properties/dimension/width.htm) '> 设置，如果自身的宽度没有定义，则长度取决于内容。

## 说明：

**复合属性。设置或检索弹性盒模型对象的子元素如何分配空间。**

* 如果缩写「flex: 1」, 则其计算值为「1 1 0%」
* 如果缩写「flex: auto」, 则其计算值为「1 1 auto」
* 如果「flex: none」, 则其计算值为「0 0 auto」
* 如果「flex: 0 auto」或者「flex: initial」, 则其计算值为「0 1 auto」，即「flex」初始值
* **示例：如下情况每个元素的计算宽是多少**

**Code：**

<ul class="flex">

<li>a</li>

<li>b</li>

<li>c</li>

</ul>

.flex{display:flex;width:800px;margin:0;padding:0;list-style:none;}

.flex :nth-child(1){flex:1 1 300px;}

.flex :nth-child(2){flex:2 2 200px;}

.flex :nth-child(3){flex:3 3 400px;}

本例定义了父容器宽（即主轴宽）为800px，由于子元素设置了伸缩基准值flex-basis，相加300+200+400=900，那么子元素将会溢出900-800=100px；  
由于同时设置了收缩因子，所以加权综合可得300\*1+200\*2+400\*3=1900px；  
于是我们可以计算a,b,c将被移除的溢出量是多少：  
a被移除溢出量：(300\*1/1900)\*100，即约等于16px  
b被移除溢出量：(200\*2/1900)\*100，即约等于21px  
c被移除溢出量：(400\*3/1900)\*100，即约等于63px  
最后a,b,c的实际宽度分别为：300-16=284px, 200-21=179px, 400-63=337px

* **仍然是上面这个例子，不过将容器的宽度改成了1500px**

**Code：**

<ul class="flex">

<li>a</li>

<li>b</li>

<li>c</li>

</ul>

.flex{display:flex;width:1500px;margin:0;padding:0;list-style:none;}

.flex :nth-child(1){flex:1 1 300px;}

.flex :nth-child(2){flex:2 2 200px;}

.flex :nth-child(3){flex:3 3 400px;}

本例定义了父容器宽（即主轴宽）为1500px，由于子元素设置了伸缩基准值flex-basis，相加300+200+400=900，那么容器将有1500-900=600px的剩余宽度；  
于是我们可以计算a,b,c将被扩展量是多少：  
a的扩展量：(1/(1+2+3))\*600，即约等于100px  
b的扩展量：(2/(1+2+3))\*600，即约等于200px  
c的扩展量：(3/(1+2+3))\*600，即约等于300px  
最后a,b,c的实际宽度分别为：300+100=400px, 200+200=400px, 400+300=700px  
从本例能看出：

当「flex-basis」在「flex」属性中不为0时（包括值为auto，此时伸缩基准值等于自身内容宽度），「flex子项」将分配容器的剩余空间（剩余空间即等于容器宽度减去各项的伸缩基准值）

当「flex-basis」在「flex」属性中等于0时，「flex子项」将分配容器的所有空间（因为各项的伸缩基准值相加等于 0，剩余空间等于容器宽度减去各项的伸缩基准值，即减0，最后剩余空间值等于容器宽度），所以可以借助此特性，给各子项定义「flex: n」来进行按比例均分容器总宽度

* 对应的脚本特性为**flex**。

