**弹性盒模型**

弹性盒子是 CSS3 的一种新的布局模式。

CSS3 弹性盒（ Flexible Box 或 flexbox），是一种当页面需要适应不同的屏幕大小以及设备类型时确保元素拥有恰当的行为的布局方式。

引入弹性盒布局模型的目的是提供一种更加有效的方式来对一个容器中的子元素进行排列、对齐和分配空白空间。

弹性盒子由弹性容器(Flex container)和弹性子元素(Flex item)组成。

弹性容器通过设置 display 属性的值为 flex将其定义为弹性容器。

弹性容器内包含了一个或多个弹性子元素。

**注意：** 弹性容器外及弹性子元素内是正常渲染的。弹性盒子只定义了弹性子元素如何在弹性容器内布局。

旧：display：box 子元素占的份数box-flex

.main{

width:100%; height: 20%; border: 1px solid #000000;

display: -webkit-box;

}

.main div{

-webkit-box-flex: 1;

text-align: center;

}

.main div:nth-child(2){

-webkit-box-flex: 8;

}

**1、box-orient**

box-orient(orient译为排列更准确)用来确定父容器里子容器的排列方式，是水平还是垂直

horizontal | vertical

**2、box-direction**

box-direction用来确定父容器里的子容器排列顺序，具体属性如下代码所示：

normal | reverse

**3、box-align**

box-align表示父容器里面子容器的垂直对齐方式，可选参数如下所示：

start | end | center | baseline | stretch

**4、box-pack**

box-pack表示父容器里面子容器的水平对齐方式，可选参数如下所示：

start | end | center | justify

弹性子元素通常在弹性盒子内一行显示。默认情况每个容器只有一行。

实例：

.main{

width: 500px;

height: 300px;

border: 1px solid #f00;

display: flex;

}

.main div{

width: 100px;

height: 100px;

background: #0ff;

}

<div class="main">

<div></div>

<div></div>

<div></div>

</div>

**效果：**



**Flex布局是什么？**

* Flex是Flexible Box的缩写，意为"弹性布局"，用来为盒状模型提供最大的灵活性。
* 任何一个容器都可以指定为Flex布局。

.box{

display: flex;

}

Webkit内核的浏览器，必须加上-webkit前缀。

.box{

display: -webkit-flex; /\* Safari \*/

display: flex;

}

注意，设为Flex布局以后，子元素的float、clear和vertical-align属性将失效。

**容器的属性**

以下6个属性设置在容器上。

* flex-direction
* flex-wrap
* flex-flow
* justify-content
* align-items
* align-content

**flex-direction 属性**

flex-direction 顺序指定了弹性子元素在父容器中的位置。

* **row：**横向从左到右排列（左对齐），默认的排列方式。
* **row-reverse：**反转横向排列（右对齐，从后往前排，最后一项排在最前面。
* **column：**纵向排列。
* **column-reverse：**反转纵向排列，从后往前排，最后一项排在最上面



**justify-content 属性**

内容对齐（justify-content）属性应用在弹性容器上，把弹性项沿着弹性容器的主轴线（main axis）对齐

* **flex-start：**

弹性项目向行头紧挨着填充。这个是默认值。第一个弹性项的main-start外边距边线被放置在该行的main-start边线，而后续弹性项依次平齐摆放。

* **flex-end：**

弹性项目向行尾紧挨着填充。第一个弹性项的main-end外边距边线被放置在该行的main-end边线，而后续弹性项依次平齐摆放。

* **center：**

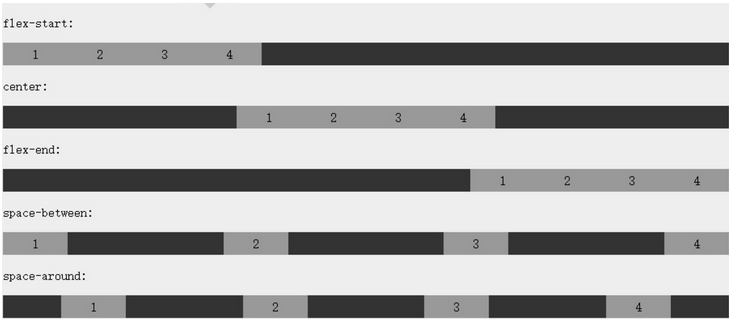
弹性项目居中紧挨着填充。（如果剩余的自由空间是负的，则弹性项目将在两个方向上同时溢出）。

* **space-between：**

弹性项目平均分布在该行上。如果剩余空间为负或者只有一个弹性项，则该值等同于flex-start。否则，第1个弹性项的外边距和行的main-start边线对齐，而最后1个弹性项的外边距和行的main-end边线对齐，然后剩余的弹性项分布在该行上，相邻项目的间隔相等。

* **space-around：**

弹性项目平均分布在该行上，两边留有一半的间隔空间。如果剩余空间为负或者只有一个弹性项，则该值等同于center。否则，弹性项目沿该行分布，且彼此间隔相等（比如是20px），同时首尾两边和弹性容器之间留有一半的间隔（1/2\*20px=10px）。



## align-items 属性

align-items 设置或检索弹性盒子元素在侧轴（纵轴）方向上的对齐方式。

* **flex-start：**弹性盒子元素的侧轴（纵轴）起始位置的边界紧靠住该行的侧轴起始边界。
* **flex-end：**弹性盒子元素的侧轴（纵轴）起始位置的边界紧靠住该行的侧轴结束边界。
* **center：**弹性盒子元素在该行的侧轴（纵轴）上居中放置。（如果该行的尺寸小于弹性盒子元素的尺寸，则会向两个方向溢出相同的长度）。
* **baseline：**如弹性盒子元素的行内轴与侧轴为同一条，则该值与'flex-start'等效。其它情况下，该值将参与基线对齐

**stretch（默认值）：**如果项目未设置高度或设为auto，将占满整个容器的高度。

。

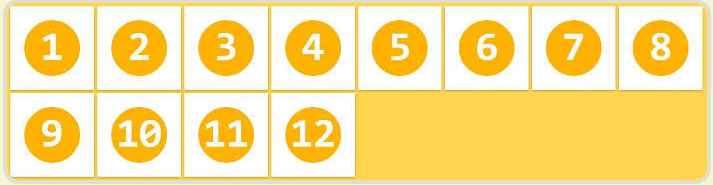
## flex-wrap 属性

**flex-wrap** 属性用于指定弹性盒子的子元素换行方式。

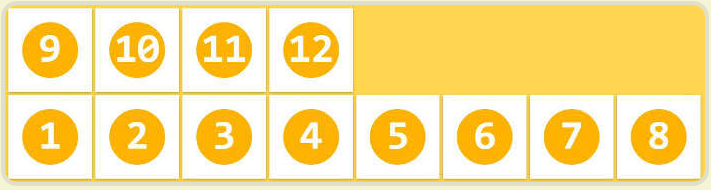
* **nowrap** - 默认， 弹性容器为单行。该情况下弹性子项可能会溢出容器。



* **wrap** - 弹性容器为多行。该情况下弹性子项溢出的部分会被放置到新行，子项内部会发生断行



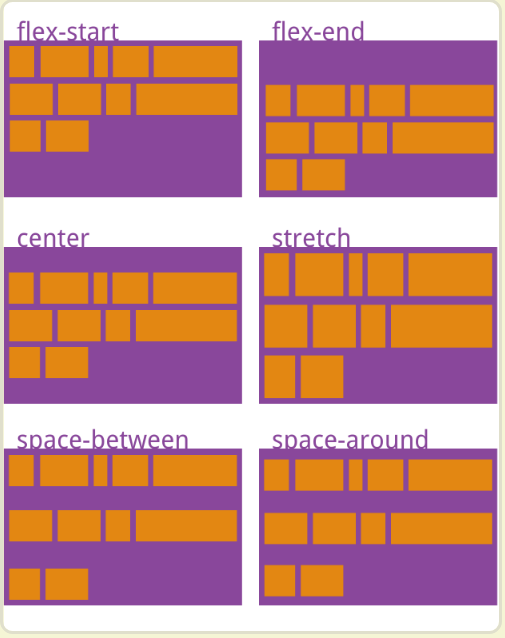
* **wrap-reverse** -反转 wrap 排列。



## align-content 属性

align-content 属性用于修改 flex-wrap 属性的行为。类似于 align-items, 但它不是设置弹性子元素的对齐，而是设置各个行的对齐。

* **stretch** - 默认。各行将会伸展以占用剩余的空间。
* **flex-start** - 各行向弹性盒容器的起始位置堆叠。
* **flex-end** - 各行向弹性盒容器的结束位置堆叠。
* **center** -各行向弹性盒容器的中间位置堆叠。
* **space-between** -各行在弹性盒容器中平均分布。
* **space-around** - 各行在弹性盒容器中平均分布，两端保留子元素与子元素之间间距大小的一半。



## 弹性子元素属性

 flex-grow定义弹性盒子元素的扩展比率。

.main div:nth-child(1){

flex-grow: 2;

}

flex-grow属性定义项目的放大比例，默认为0，即如果存在剩余空间，也不放大。

.item {

flex-grow: <number>; /\* default 0 \*/}



如果所有项目的flex-grow属性都为1，则它们将等分剩余空间（如果有的话）。如果一个项目的flex-grow属性为2，其他项目都为1，则前者占据的剩余空间将比其他项多一倍。

### align-self

align-self 属性用于设置弹性元素自身在侧轴（纵轴）方向上的对齐方式。

* auto：如果'align-self'的值为'auto'，则其计算值为元素的父元素的'align-items'值，如果其没有父元素，则计算值为'stretch'。
* flex-start：弹性盒子元素的侧轴（纵轴）起始位置的边界紧靠住该行的侧轴起始边界。
* flex-end：弹性盒子元素的侧轴（纵轴）起始位置的边界紧靠住该行的侧轴结束边界。
* center：弹性盒子元素在该行的侧轴（纵轴）上居中放置。（如果该行的尺寸小于弹性盒子元素的尺寸，则会向两个方向溢出相同的长度）。
* baseline：如弹性盒子元素的行内轴与侧轴为同一条，则该值与'flex-start'等效。其它情况下，该值将参与基线对齐。
* stretch：如果指定侧轴大小的属性值为'auto'，则其值会使项目的边距盒的尺寸尽可能接近所在行的尺寸，但同时会遵照'min/max-width/height'属性的限制。



**媒体查询：**

@media only screen and (min-width: 340px) {

body{

background: #ff0;

}

}

@media only screen and (min-width: 340px) and (max-width: 640px) {

body{

background: #f00;

}

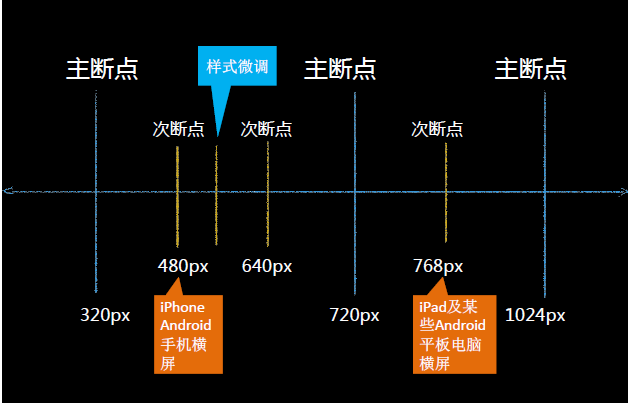
}

@media only screen and (min-width: 640px) and (max-width: 740px) {

body{

background: #0f0;

}

}

* **如何识别手机尺寸通过设置meta语句**
* **<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />**
* **device-width 设备屏幕分辨率的宽度值**
* **initial-****scale 设定页面初始缩放比例（0-10.0）**
* **minimum-scale 设定最小缩小比例（0-10.0）**
* **maximum-scale 设定最大放大比例（0-10.0）**
* **user-scalable 设定用户是否可以缩放（yes/no）**