Flexbox模型的基本使用

采用 Flex 布局的元素，称为 Flex 容器（flex container），简称"容器"。它的所有子元素自动成为容器成员，称为 Flex 项目（flex item），简称"项目"。

1. **模型中的术语**

****

1. **主轴（main axis）和侧轴（cross axis）**
2. **主轴、主轴方向：**用户代理沿着一个伸缩容器主轴配置伸缩项目，主轴是主轴方向的延伸。伸缩容器的主轴，伸缩项目主要沿着这条轴进行布局。小心，它不一定是水平的，这主要取决于justity-content属性。如果其取值为column，主轴的方向为纵向的。
3. **主轴起点、主轴终点：**伸缩项目的配置从容器的主轴起点开始，往主轴终点边结束。也就是说，伸缩项目放置在伸缩容器内从主轴起点（main start）向主轴终点（main end）的方向。
4. **主轴长度（main size）、主轴长度属性：**伸缩项目的主轴方向的宽度或高度就是项目的主轴长度，伸缩项目的主轴长度属性是width或height属性，由哪一个对着主轴方向决定。
5. **侧轴、侧轴方向：**与主轴垂直的轴叫做侧轴，侧轴是侧轴方向的延伸。主要取决于主轴方向。
6. **侧轴起点、侧轴终点：**填满项目的伸缩行的配置从容器的侧轴起点（cross start）边开始，往侧轴终点（cross end）边结束。
7. **侧轴长度（cross size）、侧轴长度属性：**伸缩项目的在侧轴方向的宽度或高度就是项目的侧轴侧轴长度，伸缩项目的侧轴长度属性是width或height属性，由哪一个对着侧轴方向决定。
8. **伸缩容器和伸缩项目**

通过display属性显示地给一个元素设置为flex或者inline-flex，这个容器就是一个伸缩容器。伸缩容器会为其内容创立新的伸缩格式化上下文，其中设置为flex的容器被渲染成一个块级元素，而设置为inline-flex的容器则渲染为一个行内元素。

若元素display的指定值是inline-flex且元素是一个浮动或绝对定位元素，则display的计算值是flex。

一个伸缩容器的内容具有零个以上的伸缩项目—伸缩容器的每个子元素（除了需要盒修复的元素之外）都会成为一个伸缩项目，且用户代理会将任何直接在伸缩容器里的连续文字块包起来成为匿名伸缩项目。

1. **伸缩容器的属性**

Flexbox伸缩布局中伸缩容器和伸缩项目是伸缩布局模块中的重要部分，其中每一部分都具有各自的属性。伸缩容器的属性包括以下几个。

1. **伸缩流方向。**是指伸缩容器的主轴方向，其决定了伸缩项目放置在伸缩容器的方向。伸缩流方向主要通过flex-direction属性设置，其默认值为row。伸缩流方向和书写模式有关系，换句话说，伸缩项目根据书写模式的方向布局。
2. **伸缩行换行。**伸缩项目在伸缩容器中有时候也会溢出伸缩容器。与传统的盒模型一样，CSS允许使用overflow属性来处理溢出内容的显示方式。

在伸缩容器中有一个伸缩换行的属性，主要用来设置伸缩容器的伸缩项目是单行显示还是多行显示，以及决定侧轴的方向。使用此属性来启用溢出的伸缩容器的伸缩项目自动换行到下一行以及控制流动方向。

在伸缩容器属性中，主要通过flex-wrap属性来设置伸缩项目是否换行，默认值为nowrap。

1. **伸缩方向与换行。**是伸缩流方向与伸缩换行的结合物，换句话说，伸缩方向与换行属性**flex-flow**同时设定了伸缩流方向flex-direction和伸缩换行flex-wrap两个属性，简而言之，是这两个属性的缩写，这两个属性决定了伸缩容器的主轴和侧轴。
2. **主轴对齐。**用来设置伸缩容器当前行伸缩项目在主轴方向的对齐的方式，指定如何在伸缩项目之间分布伸缩容器额外空间。当一行上的所伸缩项目不能伸缩或可伸缩但已达到最大长度时，这一属性才会对伸缩容器额外空间进行分配。当伸缩项目溢出某一行时，这一属性也会在项目的对齐上施加一些控制。
3. **侧轴对齐。**用来定义伸缩项目在伸缩容器的当前行的侧轴上对齐方式，这类似于主轴对齐方式，只是另一个方向。也就是说侧轴对齐主要用来指定伸缩项目在伸缩容器中如何放置和对齐的方式。
4. **堆栈伸缩行。**用来定义伸缩容器中伸缩项目行在侧轴的对齐方式，类似于侧轴对齐，只不过这是用来控制伸缩项目行在布局轴的堆放方式。
5. **伸缩项目的属性**

一个伸缩项目是一个伸缩容器的子元素。伸缩容器中的文本也被视为一个伸缩项目。伸缩项目中内容同普通流一样。比如说，当一个伸缩项目被设置为浮动，依然可以在这个伸缩项目中设置一个浮动元素。

伸缩项目都有一个主轴长度和一个侧轴长度。主轴长度是伸缩项目在主轴上的尺寸，侧轴长度是伸缩项目在侧轴上的尺寸。或者说，一个伸缩项目的宽度或高取决于伸缩容器的轴，可能就是它的主轴长度或侧轴长度。下面的几个属性可以调整伸缩项目的行为。

1. **显示顺序。**伸缩容器中的伸缩项目默认显示顺序是遵循文档在源码中的出现先后顺序（HTML文档的DOM结构中的先后顺序）。可以通过伸缩项目的显示顺序修改伸缩项目在页面中显示顺序，也可以通过这个属性对伸缩项目进行排序组合。
2. **侧轴对齐。**包括两种，一种是align-items属性，可以用来设置伸缩容器中包括匿名伸缩项目的所有项目的对齐方式；另一种是align-self属性，主要用来在单独的伸缩项目上覆写默认的对齐方式。对于匿名伸缩项目，align-self的值永远与其关联的伸缩容器的align-items的值相同。
3. **伸缩性。**定义伸缩项目可改变它们的宽度或高度填补可用的空间。可以将伸缩容器的额外空间分发给其伸缩项目（与伸缩项目的正弹性成正比）或将它们缩小以防止伸缩项目溢出（与伸缩项目的负弹性成正比）。
4. **伸缩行**

伸缩项目沿着伸缩容器内的一个伸缩行定位。伸缩容器可以是单行的，也可以是多行的。其主要由flex-wrap属性决定。单行的伸缩容器会将其所有子元素在单独的一行上进行布局，这种情况下有可能导致内容溢出伸缩容器；多行的伸缩窗口会将其伸缩项目配置在多个伸缩行上，这类似于文本的排列。当文本过宽导致一行无法容纳时，内容会断开并移至新的一行。当用户代理创建新的伸缩行时，这些伸缩行会根据flex-wrap属性沿着侧轴进行堆叠。除非伸缩容器本身是空的，每一个伸缩行至少包含一个伸缩项目。

1. **伸缩容器display**

display: flex | inline-flex

属性值flex，将一个容器设置为块伸缩容器，

属性值inline-flex，将一个容器设置为内联伸缩容器

**注意：CSS的column在伸缩容器上没有效果；float、clear和vertical-align在伸缩项目上没有效果。**

1. **伸缩流方向flex-direction**

flex-direction适用于伸缩容器，也就是伸缩项目的父元素。 主要用来创建主轴，从而定义了伸缩项目放置在伸缩容器的方向。

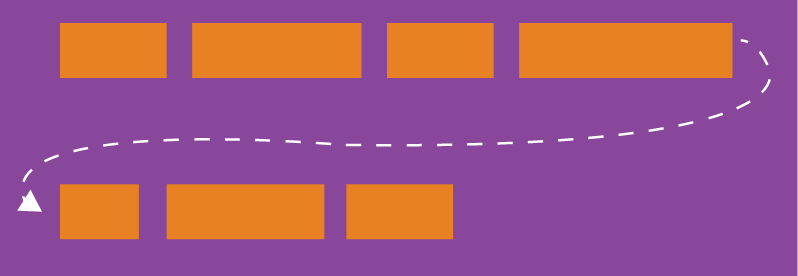
**flex-direction: row | row-reverse | column | column-reverse**

1. **row：**在ltr排版方式下从左向右排列；在rtl排版方式下从右向左排列（默认值）。
2. **row-reverse：**与row排列方式相反，在ltr排版方式下从右向左排列；在rtl排版方式下从左向右排列。
3. **column：**类似于row，不过是从上到下排列。
4. **column-reverse：**类似于row-reverse，不过是从下到上排列。



1. **伸缩换行flex-wrap**

flex-wrap属性适用于伸缩容器，也就是伸缩项目的父元素。主要用来定义伸缩容器里是单行还是多行显示，侧轴的方向决定了新行堆放的方向。



**flex-wrap：nowrap | wrap | wrap-reverse**

1. **nowrap：**伸缩容器单行显示，ltr排版下，伸缩项目从左到右排列；rtl排版下伸缩项目从右向左排列（默认值）。

****

1. **wap：**伸缩容器多行显示，ltr排版下，伸缩项目从左到右排列；rtl排版下伸缩项目从右向左排列。



1. **wap-reverse：**伸缩容器多行显示，ltr排版下，伸缩项目从右向左排列；rtl排版下，伸缩项目从左向右排列（和wrap相反）。

wap-reverse让伸缩项目在侧轴上进行start和end翻转，所以，如果伸缩项目水平排列，伸缩项目翻转不会到新的一行下面，它会翻转到一个新的行上。（简单理解就是伸缩项目只能上下或前后翻转顺序）。



1. **伸缩流方向与换行**

flex-flow适用于伸缩容器，也就是伸缩项目的父元素。这是flex-direction和flex-wrap属性的缩写版本，同时定义了伸缩容器的主轴和侧轴。其默认值为row nowrap。

**flex-flow: <’flex-direction’> || <’flex-wrap’>**

显示设置伸缩项目在主轴上的对齐方式，可以使用flex-flow属性。默认值为row，伸缩项目沿着主轴水平排列。如果想让伸缩项目沿着侧轴（垂直于主轴）排列，可以将flex-flow设置为column。当row和column属性值添加后缀“-reverse”将会反向排列（flex-flow: row-reverse或者flex-flow: column-reverse）。

另外一个大家要注意，flex-flow和writing-mode有直接的联系。当使用writing-mode: vertical-rl时转向垂直布局（如传统的中文、日文和韩文排版，也就是竖排），flex-flow: row将垂直排列伸缩项目，和column将水平排列伸缩项目。

1. **主轴对齐justy-content**

主轴对齐justy-content属性适用于伸缩容器，也就是伸缩项目的父元素。**主要用来定义伸缩项目沿着主轴线的对齐方式。**当一行的所有项目都不能伸缩或可伸缩但已到达到其最大长度时，这一属性才会对伸缩容器额外空间进行分配。当伸缩项目溢出某一行时，这一属性也会在项目的对齐上施加一些控制。

**Justify-cotent: flex-start | flex-end | center | space-between | space-around**

1. **flex-start：**伸缩项目向一行的起始位置靠齐。表示左对齐（如果书写模式为rtl，表示右对齐）
2. **flex-end：**伸缩项目向一行的结束位置靠齐。表示右对齐（如果书写模式为rtl，表示左对齐）
3. **center：**伸缩项目向一行的中间位置靠齐。
4. **space-between：**伸缩项目会平均地分布在行里。第一个伸缩项目在一行中的最开始位置，最后一个伸缩项目在一行中的最终点位置。
5. **space-around：**伸缩项目会平均地分布在行里（即均匀分布在布局轴上），两端保留一半的空间（即第一个伸缩项目前和最后一个伸缩项目后的空间为中间伸缩项目间距的一半）。

****

1. **侧轴对齐align-items和align-self**

Flexbox模型中，设置伸缩项目在侧轴的对齐方式有2种。一种是伸缩项目行在侧轴的对齐方式，主要由属性align-items控制；另一种是伸缩项目自身在侧轴的对齐方式，主要由属性align-self控制。

align-items和justify-content相呼应。align-items调整伸缩项目在侧轴上的定位方式。主要用来定义伸缩项目可以在伸缩容器的当前行的侧轴上对齐方式。 可以把它想象成侧轴（垂直于主轴）的justify-content。

align-items: flex-start | flex-end | center | baseline | stretch

1. flex-start：伸缩项目在侧轴起点边的外边距紧靠住该行在侧轴起始边。
2. flex-end：伸缩项目在侧轴终点边的外边距紧靠住该行在侧轴终点边。
3. center：伸缩项目的外边距盒在该行的侧轴上居中放置。
4. baseline：伸缩项目根据伸缩项目的基线对齐。即让伸缩项目与它们的基线对齐，基线根据伸缩项目的内容计算得到。
5. stretch：默认值，伸缩项目拉伸填充整个伸缩容器。此值会使用伸缩项目的外边距盒的尺寸在遵照（min/max-width/height）属性的限制下尽可能接近所在行的尺寸。

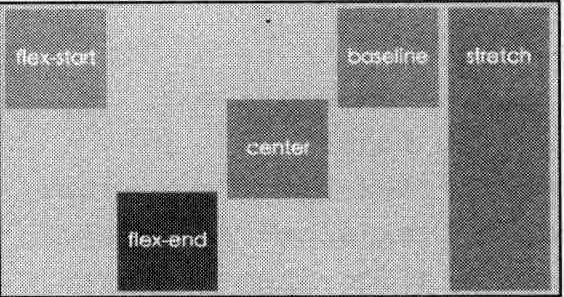
stretch会将伸缩项目从侧轴起点拉伸到侧轴终点。



伸缩项目的align-self属性主要是用来设置单独伸缩项目在侧轴的对齐方式，可以用来覆盖伸缩项目的伸缩容器的align-items属性。它的值和align-items一样。

**align-self: flex-start | flex-end | center | baseline | stretch**

对于匿名伸缩项目，align-self的值永远与其关联的伸缩容器的align-items的值相同。另外，如果伸缩项目的任一个侧轴上的外边距为auto，则align-self没有效果。如果align-self的值为auto，则其计算值为伸缩项目的伸缩容器的align-items值，如果该伸缩项目没有伸缩容器，则计算值为stretch。



1. **堆栈伸缩行align-content**

align属性主要用来调准伸缩行在伸缩容器里的对齐方式。align-content属性会更改flex-wrap的行为。它和align-items相似，但不是对齐伸缩项目，它对齐的是伸缩行。当伸缩容器的侧轴还有额外空间时，align-content属性可以用来调准伸缩行在伸缩容器里的对齐方式，这与调准伸缩项目在主轴上对齐方式的justify-content属性类似。

此属性只有伸缩项目有多行时才生效，这种情况只有display的flex-wrap为wrap时，并且没有足够的空间把伸缩项目行放在同一行中。也就是这个属性对每一行起作用而不是每个伸缩项目。

**align-content: flex-start | flex-end |center | space-between | space-around | stretch**

1. **flex-start：**各行向伸缩容器的起点位置堆叠。
2. **flex-end：**各行向伸缩容器的结束位置堆叠。
3. **center：**各行向伸缩容器的中间位置堆叠。
4. **space-between：**各行在伸缩容器中平均分布，但第一行在伸缩容器的起始位置，最后一行在伸缩容器的终点位置。
5. **space-around：**各行在伸缩容器中平均分布，两端的空间是中间空间的一半。



1. 伸缩性flex

伸缩布局决定性的特性是让伸缩项目可伸缩，也就是让伸缩项目的宽度或高度自动填充伸缩项目的额外空间，这可以用flex属性完成。

如果伸缩项目的固有宽度之和小于伸缩容器的宽度，伸缩项目会自动增加宽度以填充伸缩容器。反之，伸缩项目的固有宽度之和大于伸缩容器的宽度，伸缩项目会自动减少宽度以适应伸缩容器，不让伸缩项目溢出伸缩容器。

一个伸缩容器会等比地按照各伸缩项目的扩展比率分配额外空间，也会按照收缩比率缩小各伸缩项目以避免伸缩项目溢出伸缩容器。

**flex: none | [ <‘flex-grow’> <’flex-shrink’>? || <’flex-basis’> ]**

flex属性用来指定可伸缩长度的部件：扩展比率、收缩比率、以及伸缩基准值。当一个元素是伸缩项目时，flex属性将代替主轴长度属性决定元素的主轴长度。若元素不是伸缩项目，则flex属性没有效果。

**1）flex-grow**

举个例子，一个伸缩项目设置flex-grow值为2，另两个伸缩项目设置为flex-grow值为1，伸缩容器额外的宽度将分成4等份，设置flex-grow值为2的伸缩项目会占2份，而设置flex-grow为1的伸缩项目会占1份。

如果伸缩项目的flex-grow值都为1，则将等分伸缩容器的额外宽度。

**flex-grow: <number>** （默认值为0，即如果伸缩容器存在剩余空间，该伸缩项目也不放大）



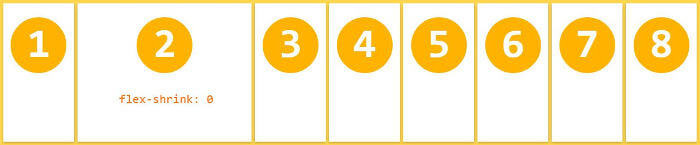
**2）flex-shrink**

flex-shrink属性适用于伸缩项目，此属性根据需要用来定义伸缩项目收缩的能力。

**flex-shrink: <number>**（默认值为1，即如果空间不足，该伸缩项目将收缩 ）

如果所有收缩项目的flex-shrink属性值都为1，当收缩容器的空间不足时，所有伸缩项目都将等比例缩小。

如果一个收缩项目的flex-shrink值为0，其他收缩项目都为1，则当收缩容器的空间不足时，前者不缩小。



1. **flex-basis**

flex-basis属性用来设置伸缩项目的伸缩基准值，伸缩容器的额外空间按比率进行伸缩。

flex-basis属性定义了在分配多余空间之前，伸缩项目占据的主轴空间（main size）。也就是根据可伸缩比率计算出额外空间的分布之前，伸缩项目主轴长度的起始值。浏览器根据这个属性，计算主轴是否有多余空间。

若在flex缩写省略此部件，flex-basis的指定值长度是0。

flex-basis的指定值为auto，则伸缩基准值的指定值是元素主轴长度属性的值，即伸缩项目本来的大小。

**flex-basis: <length> | auto;** （默认值为auto）

它可以设置为跟width或height属性一样的值（比如350px），即项目将占据固定空间。

1. **flex**

flex属性是flex-grow，flex-shrink和flex-basis的简写，默认值为0 1 auto。后两个属性可选。

**flex: none | [ <’flex-grow’> <’flex-shrink’>? || <’flex-basis’> ]**

该属性有两个快捷值：auto（1 1 auto）和none（0 0 auto）。

建议优先使用这个属性，而不是单独写三个分离的属性，因为浏览器会推算相关值。

a、flex:0 auto和flex:initial与flex:0 1 auto相同。这也就是flex的初始值。根据width/height属性决定元素的尺寸。