# css学习第一部分，查看：[css学习笔记1](学习笔记05-css学习1.docx)

# css学习第二部分，查看：[css学习笔记2](学习笔记05-css学习2.docx)

# css学习第三部分：

# 接着上一部分，我们来继续学习

# 13.dev学习项目，项目初始状态只是纯html，你需要把它变为响应式网页

## 练习8.4，需要多多练习

## 这个项目主要学习前面的float和媒体查询，solution代码

|  |
| --- |
| body {    font-family: "Poppins", sans-serif;    margin: 50px 50px 0 50px;    background-color: #faf9f6;    display: flex;    flex-direction: column;    min-height: 95vh;  }  .main {    flex: 1;  }  h1 {    font-size: 5rem;  }  footer {    text-align: right;    color: midnightblue;  }  .tile-image {    height: 200px;    float: left;    margin-right: 50px;  }  .card {    width: 45%;  }  .left {    float: left;  }  .right {    float: right;  }  .creative {    color: midnightblue;  }  @media (max-width: 680px) {    .logo {      width: 100px;    }    h1 {      font-size: 3.5rem;      text-align: center;    }    .card {      width: 100%;      display: block;      margin-bottom: 30px;      text-align: justify;    }    .card img {      margin-bottom: 10px;      width: 100%;      display: inline;      object-fit: cover;    }  } |

## html代码

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>    <meta charset="UTF-8">    <meta name="viewport" content="width=device-width, initial-scale=1.0">    <title>Agency</title>    <link rel="stylesheet" href="./style.css">    <link rel="preconnect" href="https://fonts.googleapis.com">    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>    <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;700&display=swap" rel="stylesheet">  </head>  <body>    <div class="main">      <img class="logo" src="./assets/images/logo.png" alt="logo">      <h1>We are a <span>Creative</span> Design Agency</h1>      <div class="left card"><img class="tile-image" src="./assets/images/beautiful.jpg" alt="hand and flower in water">        <h2 class="card-title">Beauty</h2>        <p class="card-text">We strive to create the most beautiful websites for all your needs. Working closely with you          to          design and          develop an amazing website for your business.</p>      </div>      <div class="right card"><img class="tile-image" src="./assets/images/construction.jpg" alt="metal structure">        <h2 class="card-title">Construction</h2>        <p>Built by our team of professional developers, we ensure the most rigourous and modern websites. Built from          scratch using HTML and CSS. Only the best for you.</p>      </div>    </div>    <footer>      <p>Create. Develop. Design.</p>    </footer>  </body>  </html> |

# 14、Diaplsy flex弹性布局

## 弹性盒子是用来做布局的比较好的选择，现代web开发以及不使用float来做布局了，float仅仅用来做文字环绕的效果。也可以在布局的一部分里面使用float。建议使用网格，弹性盒子或者bootstrap来做布局。

## 需要注意的是，在弹性盒子布局里面，最好不要使用以前的inline、block、inline-block或者none属性。因为flex以及不需要使用他们。你还是可以在别的布局里面使用他们。我们知道块级元素默认的显示属性是block，span等等行内元素默认的显示属性是inline，img元素的默认显示属性是inline-block.但是，如果我们给他们的父元素使用display:flex；后，父元素就是一个弹性盒子把他们全部包裹，他们原来的显示属性全部失效，必须使用弹性显示属性。默认弹性盒子里面元素一个一个一个紧紧挨着的，你需要设置gap属性来设置间隔。

## 默认flex容器的显示属性是block，你可以设置display:inline-flex，这样子弹性盒子容器的显示属性就有点像inline-block的效果，注意：这里指的的父容器，容器里面的元素是弹性布局属性

## 设置弹性布局的需要如下步骤

## 1.需要将父容器的里面设置display:flex;然后设置gap，否则所有子元素否粘在一起

|  |
| --- |
|  |

## 2.设置弹性布局方向

### 1）flex-direction:row :所有子元素排成一行，这是弹性盒子默认方向属性

|  |
| --- |
|  |

#### 注意：这个设置的main-axis是水平方向，cross-axis是垂直方向

### 2）flex-direction: column：所有子元素排成一列

|  |
| --- |
|  |

#### 注意：这个设置的main-axis是垂直方向，cross-axis是水平方向

## 3.flex-basis，需要根据主轴来进行。这个属性是设置为子元素的，不是容器

### column

|  |
| --- |
|  |

### row

|  |
| --- |
|  |

### 所以说，搞清楚主轴在哪个方向也是挺重要的

## 4.弹性布局的实现

### 1）顺序order，就是子元素的排列顺序，默认值是0，弹性盒子的方向是row时，值越大越靠右边。修改这个值就可以改变排列顺序，注意是加在子元素上面的

### 2）flex-wrap属性，添加到父容器的属性

#### 默认情况下，方向为row的弹性布局是不换行的。当元素很多放不下，元素会超出容器，非常难看，如果需要换行就使用flex-wrap: wrap;前提是要设置display:flex;

### 实例

使弹性项目在需要时换行：

div {

display: flex;

flex-wrap: wrap;

}

### CSS 语法

flex-wrap: nowrap|wrap|wrap-reverse|initial|inherit;

### 属性值

|  |  |
| --- | --- |
| **值** | **描述** |
| nowrap | 默认值。规定弹性项目不会换行。 |
| wrap | 规定弹性项目会在需要时换行。 |
| wrap-reverse | 规定弹性项目会在需要时换行，以反方向。 |
| initial | 将此属性设置为其默认值。参阅 [initial](https://www.w3school.com.cn/cssref/css_initial.asp)。 |
| inherit | 从其父元素继承此属性。参阅 [inher](https://www.w3school.com.cn/cssref/css_inherit.asp) |

### 3）Justify-content：添加到父容器中

### 定义和用法

justify-content 用于设置或检索弹性盒子元素在主轴（横轴）方向上的对齐方式。

**提示：**使用 align-content 属性对齐交叉轴上的各项（垂直）。

|  |  |
| --- | --- |
| **默认值：** | flex-start |
| **继承：** | 否 |
| **可动画化：** | 否。请参阅 [可动画化（animatable）](https://www.runoob.com/cssref/css-animatable.html)。 |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.justifyContent="space-between"[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss3_js_justify-content) |

*/\* 对齐方式 \*/*  
**justify-content**: center;     */\* 居中排列 \*/*  
**justify-content**: start;      */\* 从行首开始排列 \*/*  
**justify-content**: end;        */\* 从行尾开始排列 \*/*  
**justify-content**: flex-start; */\* 从行首起始位置开始排列 \*/*  
**justify-content**: flex-end;   */\* 从行尾位置开始排列 \*/*  
**justify-content**: **left**;       */\* 一个挨一个在对齐容器得左边缘 \*/*  
**justify-content**: **right**;      */\* 元素以容器右边缘为基准，一个挨着一个对齐, \*/*  
  
*/\* 基线对齐 \*/*  
**justify-content**: baseline;  
**justify-content**: first baseline;  
**justify-content**: last baseline;  
  
*/\* 分配弹性元素方式 \*/*  
**justify-content**: space-between;  */\* 均匀排列每个元素  
                                   首个元素放置于起点，末尾元素放置于终点 \*/*  
**justify-content**: space-around;  */\* 均匀排列每个元素  
                                   每个元素周围分配相同的空间 \*/*  
**justify-content**: space-evenly;  */\* 均匀排列每个元素  
                                   每个元素之间的间隔相等 \*/*  
**justify-content**: stretch;       */\* 均匀排列每个元素  
                                   'auto'-sized 的元素会被拉伸以适应容器的大小 \*/*  
  
*/\* 溢出对齐方式 \*/*  
**justify-content**: safe center;  
**justify-content**: unsafe center;  
  
*/\* 全局值 \*/*  
**justify-content**: inherit;  
**justify-content**: initial;  
**justify-content**: unset;

### CSS 语法

justify-content: flex-start|flex-end|center|space-between|space-around|initial|inherit;

### 属性值

|  |  |  |
| --- | --- | --- |
| **值** | **描述** | **测试** |
| flex-start | 默认值。从行首起始位置开始排列。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=flex-start) |
| flex-end | 从行尾位置开始排列。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=flex-end) |
| center | 居中排列。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=center) |
| space-between | 均匀排列每个元素，首个元素放置于起点，末尾元素放置于终点。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=space-between) |
| space-evenly | 均匀排列每个元素，每个元素之间的间隔相等。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=space-evenly) |
| space-around | 均匀排列每个元素，每个元素周围分配相同的空间。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=space-around) |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_justify-content&preval=initial) |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |  |

### 效果实例

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

#### space-between和space-around的区别，space-between会保证第一个元素紧贴左边。最后一个元素紧贴右边（如果方向是row）而space-around就是所有地方都需要有空白包括最左边和最右边。space-evenly在space-around的基础上把所有空白的文字设置相同

### 4）align-items，子元素的对齐方式

### 定义和用法

align-items 属性定义flex子项在flex容器的当前行的侧轴（纵轴）方向上的对齐方式。

**提示：**使用每个弹性对象元素的 align-self 属性可重写 align-items 属性。

|  |  |
| --- | --- |
| **默认值：** | stretch |
| **继承：** | 否 |
| **可动画化：** | 否。请参阅 [可动画化（animatable）](https://www.runoob.com/cssref/css-animatable.html)。 |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.alignItems="center"[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss3_js_align-items) |

### CSS 语法

align-items: stretch|center|flex-start|flex-end|baseline|initial|inherit;

### 属性值

|  |  |  |
| --- | --- | --- |
| **值** | **描述** | **测试** |
| stretch | 默认值。元素被拉伸以适应容器。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-items&preval=stretch) |
| center | 元素位于容器的中心。  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）。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-items&preval=center) |
| flex-start | 元素位于容器的开头。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-items&preval=flex-start) |
| flex-end | 元素位于容器的结尾。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-items&preval=flex-end) |
| baseline | 元素位于容器的基线上。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-items&preval=baseline) |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-items&preval=initial) |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |  |

### 效果展示，注意设置align-items属性之前，需要设置flex-wrap：nowrap；

|  |
| --- |
|  |
|  |
| 拉伸 |

### 5）align-self，添加到弹性容器的子元素上

### 定义和用法

align-self 属性定义flex子项单独在侧轴（纵轴）方向上的对齐方式。

**注意：**align-self 属性可重写灵活容器的 align-items 属性。

### CSS 语法

align-self: auto|stretch|center|flex-start|flex-end|baseline|initial|inherit;

### 属性值

|  |  |  |
| --- | --- | --- |
| **值** | **描述** | **测试** |
| auto | 默认值。元素继承了它的父容器的 align-items 属性。如果没有父容器则为 "stretch"。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=auto) |
| stretch | 元素被拉伸以适应容器。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=stretch) |
| center | 元素位于容器的中心。  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）。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=center) |
| flex-start | 元素位于容器的开头。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=flex-start) |
| flex-end | 元素位于容器的结尾。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=flex-end) |
| baseline | 元素位于容器的基线上。  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。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=baseline) |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-self&preval=initial) |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |  |

### 效果展示

|  |
| --- |
|  |

### 6）align-content,这个属性添加到父容器上，也就是弹性盒子

### 定义和用法

align-content 属性在弹性容器内的各项没有占用交叉轴上所有可用的空间时对齐容器内的各项（垂直）。

**提示：**使用 justify-content 属性对齐主轴上的各项（水平）。

**注意：**容器内必须有多行的项目，该属性才能渲染出效果。

|  |  |
| --- | --- |
| **默认值：** | stretch |
| **继承：** | 否 |
| **可动画化：** | 否。请参阅 [*可动画化（animatable）*](https://www.runoob.com/cssref/css-animatable.html)。 |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.alignContent="center" |

### CSS 语法

align-content: stretch|center|flex-start|flex-end|space-between|space-around|initial|inherit;

### 默认值

|  |  |  |
| --- | --- | --- |
| **值** | **描述** | **测试** |
| stretch | 默认值。元素被拉伸以适应容器。  各行将会伸展以占用剩余的空间。如果剩余的空间是负数，该值等效于'flex-start'。在其它情况下，剩余空间被所有行平分，以扩大它们的侧轴尺寸。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=stretch) |
| center | 元素位于容器的中心。  各行向弹性盒容器的中间位置堆叠。各行两两紧靠住同时在弹性盒容器中居中对齐，保持弹性盒容器的侧轴起始内容边界和第一行之间的距离与该容器的侧轴结束内容边界与第最后一行之间的距离相等。（如果剩下的空间是负数，则各行会向两个方向溢出的相等距离。） | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=center) |
| flex-start | 元素位于容器的开头。  各  行向弹性盒容器的起始位置堆叠。弹性盒容器中第一行的侧轴起始边界紧靠住该弹性盒容器的侧轴起始边界，之后的每一行都紧靠住前面一行。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=flex-start) |
| flex-end | 元素位于容器的结尾。  各行向弹性盒容器的结束位置堆叠。弹性盒容器中最后一行的侧轴起结束界紧靠住该弹性盒容器的侧轴结束边界，之后的每一行都紧靠住前面一行。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=flex-end) |
| space-between | 元素位于各行之间留有空白的容器内。  各行在弹性盒容器中平均分布。如果剩余的空间是负数或弹性盒容器中只有一行，该值等效于'flex-start'。在其它情况下，第一行的侧轴起始边界紧靠住弹性盒容器的侧轴起始内容边界，最后一行的侧轴结束边界紧靠住弹性盒容器的侧轴结束内容边界，剩余的行则按一定方式在弹性盒窗口中排列，以保持两两之间的空间相等。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=space-between) |
| space-around | 元素位于各行之前、之间、之后都留有空白的容器内。  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。如果剩余的空间是负数或弹性盒容器中只有一行，该值等效于'center'。在其它情况下，各行会按一定方式在弹性盒容器中排列，以保持两两之间的空间相等，同时第一行前面及最后一行后面的空间是其他空间的一半。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=space-around) |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 | [**测试 »**](https://www.runoob.com/try/playit.php?f=playcss_align-content&preval=initial) |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |  |

### 效果展示：

|  |
| --- |
|  |

#### 注意：要align-cotent生效，弹性盒子必须设置flex-wrap：wrap；

### 7）flex-flow

### CSS flex-flow 属性

[[CSS 参考手册](https://www.runoob.com/cssref/css-reference.html) CSS 参考手册](https://www.runoob.com/cssref/css-reference.html)

### 实例

让弹性盒的元素以相反的顺序显示，且在必要的时候进行拆行：

display:flex;  
flex-flow:row-reverse wrap;

### 定义和用法

flex-flow 属性是 flex-direction 和 flex-wrap 属性的复合属性。

flex-flow 属性用于设置或检索弹性盒模型对象的子元素排列方式。

flex-direction 属性规定灵活项目的方向。

flex-wrap 属性规定灵活项目是否拆行或拆列。

**注意：**如果元素不是弹性盒对象的元素，则 flex-flow 属性不起作用。

|  |  |
| --- | --- |
| **默认值：** | row nowrap |
| **继承：** | 否 |
| **可动画化：** | 否。请参阅 [可动画化（animatable）](https://www.runoob.com/cssref/css-animatable.html)。 |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.flexFlow="column nowrap"[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss3_js_flex-flow) |

### CSS 语法

flex-flow: *flex-direction* *flex-wrap*|initial|inherit;

### 属性值

|  |  |
| --- | --- |
| **值** | **描述** |
| flex-direction | 可能的值：  row row-reverse column column-reverse initial inherit  默认值是 "row"。  规定灵活项目的方向。 |
| flex-wrap | 可能的值：  nowrap wrap wrap-reverse initial inherit  默认值是 "nowrap"。  规定灵活项目是否拆行或拆列。 |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |

### 8)flex-shrink弹性收缩

#### 实例：

|  |
| --- |
| **<!DOCTYPE html>**  **<html>**  **<head>**  **<meta charset="utf-8">**  **<title>菜鸟教程(runoob.com)</title>**  **<style>**  **#content {**  **display: flex;**  **width: 500px;**  **}**  **#content div {**  **flex-basis: 120px;**  **border: 3px solid rgba(0,0,0,.2);**  **}**  **.box {**  **flex-shrink: 1;**  **}**  **.box1 {**  **flex-shrink: 2;**  **}**  **</style>**  **</head>**  **<body>**  **<p>div 总宽度为 500px, flex-basic 为 120px。</p>**  **<p>A, B, C 设置 flex-shrink:1。 D , E 设置为 flex-shrink:2</p>**  **<p>D , E 宽度与 A, B, C 不同</p>**  **<div id="content">**  **<div class="box" style="background-color:red;">A</div>**  **<div class="box" style="background-color:lightblue;">B</div>**  **<div class="box" style="background-color:yellow;">C</div>**  **<div class="box1" style="background-color:brown;">D</div>**  **<div class="box1" style="background-color:lightgreen;">E</div>**  **</div>**  **<p><b>注意:</b> Internet Explorer 10 及更早版本浏览器不支持 flex-shrink 属性。</p>**  **<p><b>注意:</b> Safari 6.1 及更新版本通过 -webkit-flex-shrink 属性支持该属性。</p>**  **</body>**  **</html>** |

#### 效果：

|  |
| --- |
|  |

**实例解析：**

flex-shrink的默认值为1，如果没有显示定义该属性，将会自动按照默认值1在所有因子相加之后计算比率来进行空间收缩。

本例中A、B、C 显式定义了 flex-shrink 为 1，D、E 定义了 flex-shrink 为 2，所以计算出来总共将剩余空间分成了 7 份，其中 A、B、C 占 1 份，D、E 占 2 份，即1:1:1:2:2

我们可以看到父容器定义为 500px，子项被定义为 120px，子项相加之后即为 600 px，超出父容器 100px。那么超出的 100px 需要被 A、B、C、D、E 消化 通过收缩因子，所以加权综合可得 **100\*1+100\*1+100\*1+100\*2+100\*2=700px**。

于是我们可以计算 A、B、C、D、E 将被移除的溢出量是多少：

A 被移除溢出量：(100\*1/700)\*100，即约等于14px

B 被移除溢出量：(100\*1/700)\*100，即约等于14px

C 被移除溢出量：(100\*1/700)\*100，即约等于14px

D 被移除溢出量：(100\*2/700)\*100，即约等于28px

E 被移除溢出量：(100\*2/700)\*100，即约等于28px

最后A、B、C、D、E的实际宽度分别为：**120-14=106px, 120-14=106px, 120-14=106px, 120-28=92px,120-28=92px**，此外，这个宽度是包含边框的。

### 定义和用法

flex-shrink 属性指定了 flex 元素的收缩规则。flex 元素仅在默认宽度之和大于容器的时候才会发生收缩，其收缩的大小是依据 flex-shrink 的值。

**注意：**如果元素不是弹性盒对象的元素，则 flex-shrink 属性不起作用。

|  |  |
| --- | --- |
| **默认值：** | 1 |
| **继承：** | 否 |
| **可动画化：** | 是。请参阅 [可动画化（animatable）](https://www.runoob.com/cssref/css-animatable.html)。[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss_anim_flex-shrink) |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.flexShrink="5"[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss3_js_flex-shrink) |

### CSS 语法

flex-shrink: *number*|initial|inherit;

### 属性值

|  |  |
| --- | --- |
| **值** | **描述** |
| number | 一个数字，规定项目将相对于其他灵活的项目进行收缩的量。默认值是 1。 |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |

### 课堂实例

|  |
| --- |
|  |

### 如果我们把上面的例子的容器改为display：flex;默认里面的所有子元素就会按照行排列而且一个紧挨着一个。如果我们需要修改他们的宽度，它有一个优先级如下

|  |
| --- |
|  |

### 它是从右到左优先级逐渐降低，也就是先看看有没有设置最大或者最小宽度(如果flex方向是column它指的是高度)，如果没有就看看有没有设置flex-basis然后是元素的宽度最后才是内容宽度，下面我们来看看一些效果

|  |
| --- |
|  |

### 此时，如果我们把窗口宽度缩小，效果如下

|  |
| --- |
|  |

### 你不断收窄窗口宽度，他会一直缩小，直到极限

|  |
| --- |
|  |

### 这样子有些内容就看不见了

|  |
| --- |
|  |

### 但是你把窗口的宽度拉大，他们一开始会慢慢变宽，但是当他们回复原来的大小后就不再变宽，下面就是最大宽度和最小宽度的效果

|  |
| --- |
|  |

### 给子元素设置宽度的效果

|  |
| --- |
|  |

### 把窗口收窄元素会跟着缩小，如果容器的大小缩小到比所有元素的总宽度还小，元素最窄可以收窄到内容的宽度，效果如下

|  |
| --- |
|  |

### 如果此时你有设置了flex-basis，你会发现它比width优先，元素宽度变为flex-basis指定的宽度

|  |
| --- |
|  |

### 此时如果我们收窄窗口宽度，当窗口宽度不够容纳他们原来的尺寸，他们的宽度会慢慢缩小直到极限

|  |
| --- |
|  |
|  |

### 此时如果你设置了最大宽度和flex-basics,它会取这两个值中比较小的那个。

|  |
| --- |
|  |

### 最大宽度指的是元素最大只能够达到这个宽度，以后就不能够再增加。

|  |
| --- |
|  |

### 同理，最小宽度是指元素最小能够缩小到这个宽度，以后就不能够再缩小

### 再来看看这个例子，如果我们这么写，效果如何？

|  |
| --- |
|  |

### 效果是：所有元素都变为300px宽，因为min-width优先级比flex-basic高

|  |
| --- |
|  |

### 此时即使你缩小窗口宽度，元素的宽度不会最缩小，他只是会隐藏超出的部分

|  |
| --- |
|  |

### 9）flex-grow

#### 实例

|  |
| --- |
| **<!DOCTYPE html>**  **<html>**  **<head>**  **<style>**  **#main {**  **width: 350px;**  **height: 100px;**  **border: 1px solid #c3c3c3;**  **display: flex;**  **}**  **#main div:nth-of-type(1) {flex-grow: 1;}**  **#main div:nth-of-type(2) {flex-grow: 3;}**  **#main div:nth-of-type(3) {flex-grow: 1;}**  **#main div:nth-of-type(4) {flex-grow: 1;}**  **#main div:nth-of-type(5) {flex-grow: 1;}**  **</style>**  **</head>**  **<body>**  **<div id="main">**  **<div style="background-color:coral;"></div>**  **<div style="background-color:lightblue;"></div>**  **<div style="background-color:khaki;"></div>**  **<div style="background-color:pink;"></div>**  **<div style="background-color:lightgrey;"></div>**  **</div>**  **<p><b>注意:</b> Internet Explorer 10 及更早版本浏览器不支持 flex-grow 属性。</p>**  **<p><b>注意:</b> Safari 6.1 及更新版本通过 -webkit-flex-grow 属性支持该属性。</p>**  **</body>**  **</html>** |

#### 效果

|  |
| --- |
|  |

### 定义和用法

flex-grow 属性用于设置或检索弹性盒子的扩展比率。。

**注意：**如果元素不是弹性盒对象的元素，则 flex-grow 属性不起作用。

|  |  |
| --- | --- |
| **默认值：** | 0 |
| **继承：** | 否 |
| **可动画化：** | 是。请参阅 [可动画化（animatable）](https://www.runoob.com/cssref/css-animatable.html)。[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss_anim_flex-grow) |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.flexGrow="5"[**尝试一下**](https://www.runoob.com/try/try.php?filename=trycss3_js_flex-grow) |

### CSS 语法

flex-grow: *number*|initial|inherit;

### 属性值

|  |  |
| --- | --- |
| **值** | **描述** |
| number | 一个数字，规定项目相对于其他灵活的项目进行扩展的量。默认值是 0。 |
| initial | 设置该属性为它的默认值。请参阅 [initial](https://www.runoob.com/cssref/css-initial.html)。 |
| inherit | 从父元素继承该属性。请参阅 [inherit](https://www.runoob.com/cssref/css-inherit.html)。 |

### 需要注意：flex-grow和flex-shrink是很重要的，我们看看下面的例子

|  |
| --- |
|  |

### 此时你增加绘制缩小容器的宽度，元素的宽度都不会变化因为你把放大是缩小的增量都设置为0，也就是不会变化

|  |
| --- |
|  |

### 如果你把flex-grow设置为1，flex-shrink设置为0，元素的宽度先会自动增长并且平分容器的宽度

|  |
| --- |
|  |

### 当容器的宽度缩小，元素的宽度也跟着缩小直到flex-basis指定的宽度，然后就不再缩小

|  |
| --- |
|  |

## 如果我们发过来，把flex-grow改为0，flex-shrink改为1，默认效果如下

|  |
| --- |
|  |

### 元素的默认宽度在flex-basis指定的宽度，此时如果我们增加容器宽度，元素的宽度不会增加，我们缩小容器的宽度，元素的宽度会一直缩小到极限

|  |
| --- |
|  |

### 这个其实就是默认值，那么如果两个值都设置为1呢，如图

|  |
| --- |
|  |

### 此时其实就是上面2种情况的结合，不过此时flex-basis属性完全被忽略，元素默认会占用所有容器的宽度然后平分，缩小的时候会一直缩小到极限

|  |
| --- |
|  |

## 其实flex-basis可以使用auto或者0，auto会根据元素的内容来给他分配宽度这样子使用的元素的宽度可能不一样，设置为0就使得他们宽度一样，可以把他们三个简写如下

|  |  |  |
| --- | --- | --- |
|  |  |  |

### 注意：flex-basis需要写在最后，书写顺序是grow shrink basis；然后还可以再简写为flex:1;

### 实例

|  |
| --- |
|  |

### Flex:2;是值flex-grow:2;flex-shrink:2;flex-basis:0;

## 这里老师有一个练习网站：[Flexbox Sizing Exercise (appbrewery.github.io)](https://appbrewery.github.io/flexbox-sizing-exercise/)

### 可以慢慢练习

### 参考答案

|  |
| --- |
|  |

# css flex指南网站

## [A Complete Guide To Flexbox | CSS-Tricks](https://css-tricks.com/snippets/css/a-guide-to-flexbox/)

## 弹性盒子练习网站有很多，这里有一个比较好的：[Flexbox Froggy - A game for learning CSS flexbox](https://flexboxfroggy.com/)