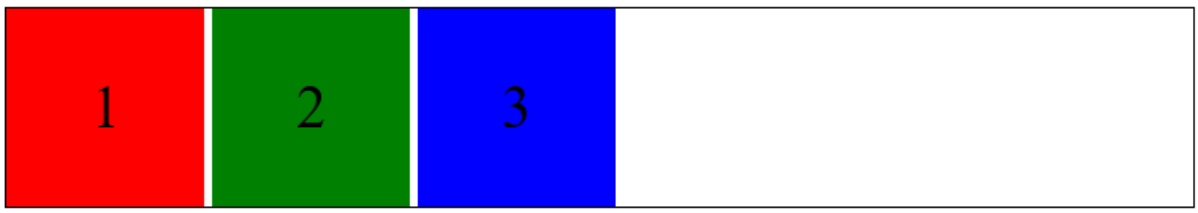
[笔记][LIKE-C3][02-伸缩布局]

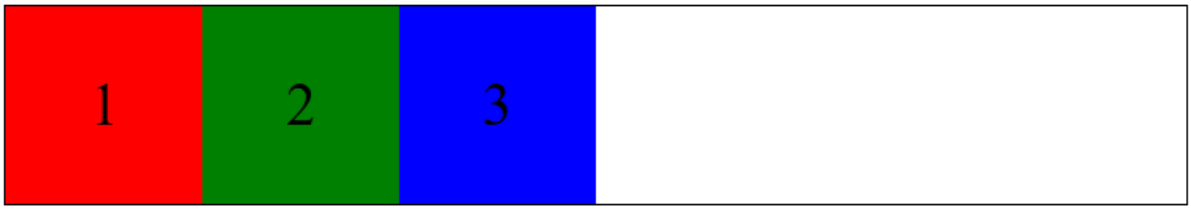
前端

17. 体验伸缩布局(掌握)



删除间隙：

- 去掉 `div` 的换行（不利于代码阅读）
- 使用浮动，但是浮动的元素不会撑起父元素的高度，所以要清浮动。找到父元素，然后 `overflow: hidden;`



还可以使用伸缩布局

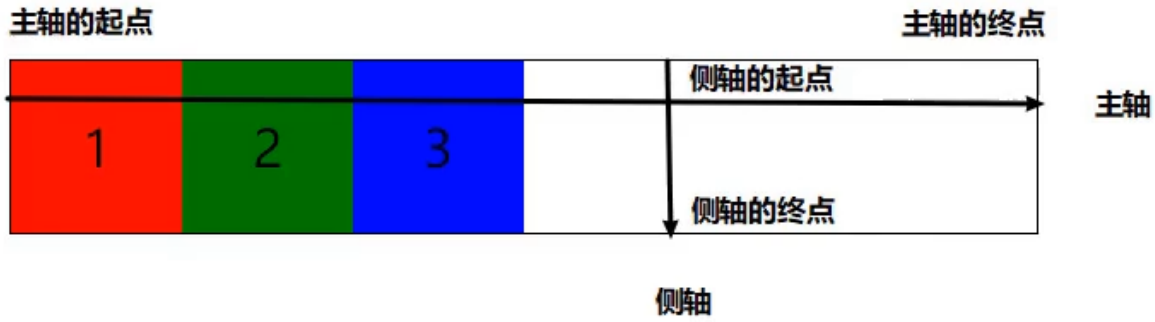
> 快捷键 `df` 相当于 `display: flex;`

伸缩布局比浮动简单：

- 浮动：子元素浮动，父元素清浮动
- 伸缩布局：父元素伸缩布局

伸缩容器
被设置了 `display: flex;` 的元素就是伸缩容器。

伸缩项
伸缩容器中的子元素就是伸缩项。



主轴

- 默认是水平方向的轴
- 主轴起点在伸缩容器的最左边
- 主轴终点在伸缩容器的最右边

侧轴

- 默认是垂直方向的轴
- 侧轴起点在伸缩容器的最顶部
- 侧轴终点在伸缩容器的最底部

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
```

```
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            /* overflow: hidden; */
            display: flex;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
            background: red;
            /* display: inline-block; */
            /* float: left; */
        }

        ul>li:nth-child(2) {
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>
<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>
</html>
```

18. 伸缩布局-主轴方向(掌握)

团子注：

- 老师这里讲的 `flex-direction` 我觉的理解成**伸缩方向**比较好，见名知意。
- 不用理解成主轴起点。

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
            /*
            在伸缩布局中，默认情况下水平方向是主轴。
            主轴起点默认在伸缩容器的最左边。
            默认情况下所有的伸缩项都是从主轴的起点开始排版的。
            但是我们也可以通过修改属性来修改主轴的起点位置。

            flex-direction 用于修改主轴起点的位置
            row：起点在伸缩容器的最左边，终点在伸缩容器的最右边，从左至右排版，是默认
的取值
            row-reverse：起点在伸缩容器的最右边，终点在伸缩容器的最左边。
            column：起点在伸缩容器的最顶部，终点在伸缩容器的最底部，从上至下排版
            column-reverse：起点在伸缩容器的最底部，终点在伸缩容器的最顶部，从下至
上排版

            注意点：
                在伸缩布局中，主轴和侧轴永远都是十字交叉的
                只要主轴的方向发生了变化，侧轴也会发生变化。
             */
            flex-direction: row;
            flex-direction: row-reverse;
            flex-direction: column;
            flex-direction: column-reverse;
        }

        ul>li {
```

```
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
            background: red;
        }

        ul>li:nth-child(2) {
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

19. 伸缩布局-主轴对齐方式(掌握)

掌握了伸缩布局的**方向**，下面学习**对齐方式**

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }
```

```
        ul {
            list-style: none;
            width: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
            /*
                flex-direction: row; 是主轴起点的默认值
                justify-content: flex-start; 主轴上伸缩项对齐的默认值
            */
            flex-direction: row;
            justify-content: flex-start;

            /* 与主轴终点对齐 */
            /* 注意与 flex-direction: row-reverse; 顺序不同 */
            justify-content: flex-end;

            /* 与主轴中点对齐 */
            justify-content: center;

            /* 两端对齐 */
            /* (伸缩容器主轴宽度 - 伸缩项宽度) / (伸缩项个数 - 1) */
            /* 把上面计算出来的空间放到伸缩项之间 */
            justify-content: space-between;

            /* 环绕对齐（两端留白） */
            /* (伸缩容器主轴宽度 - 伸缩项宽度) / (伸缩项个数 * 2) */
            /* 左右两端填充上面计算出来的一个宽度，伸缩项之间填充两个宽度 */
            justify-content: space-around;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
            background: red;
        }

        ul>li:nth-child(2) {
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>

<body>
    <ul>
```

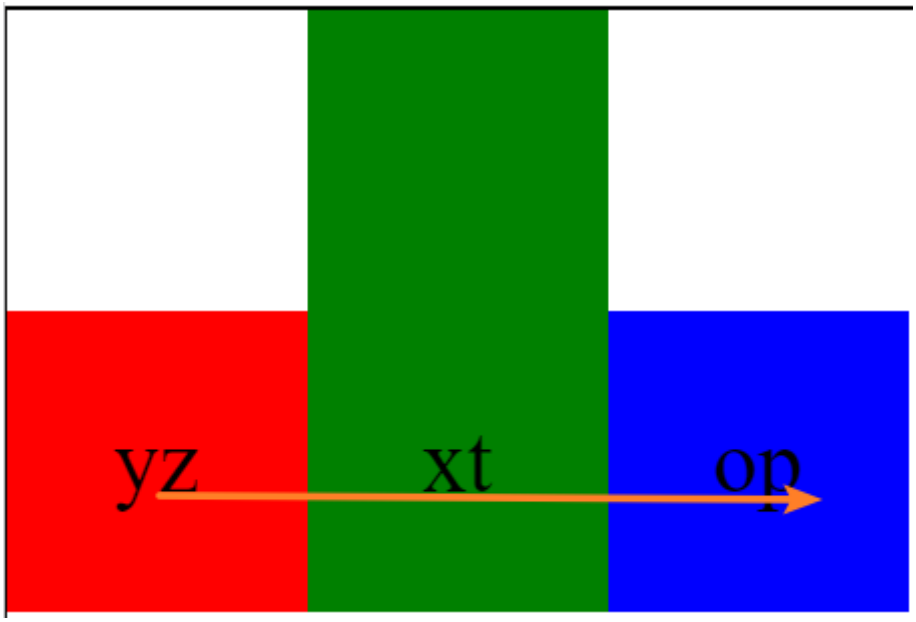
```
            <li>1</li>
            <li>2</li>
            <li>3</li>
        </ul>
    </body>

    </html>
```

---

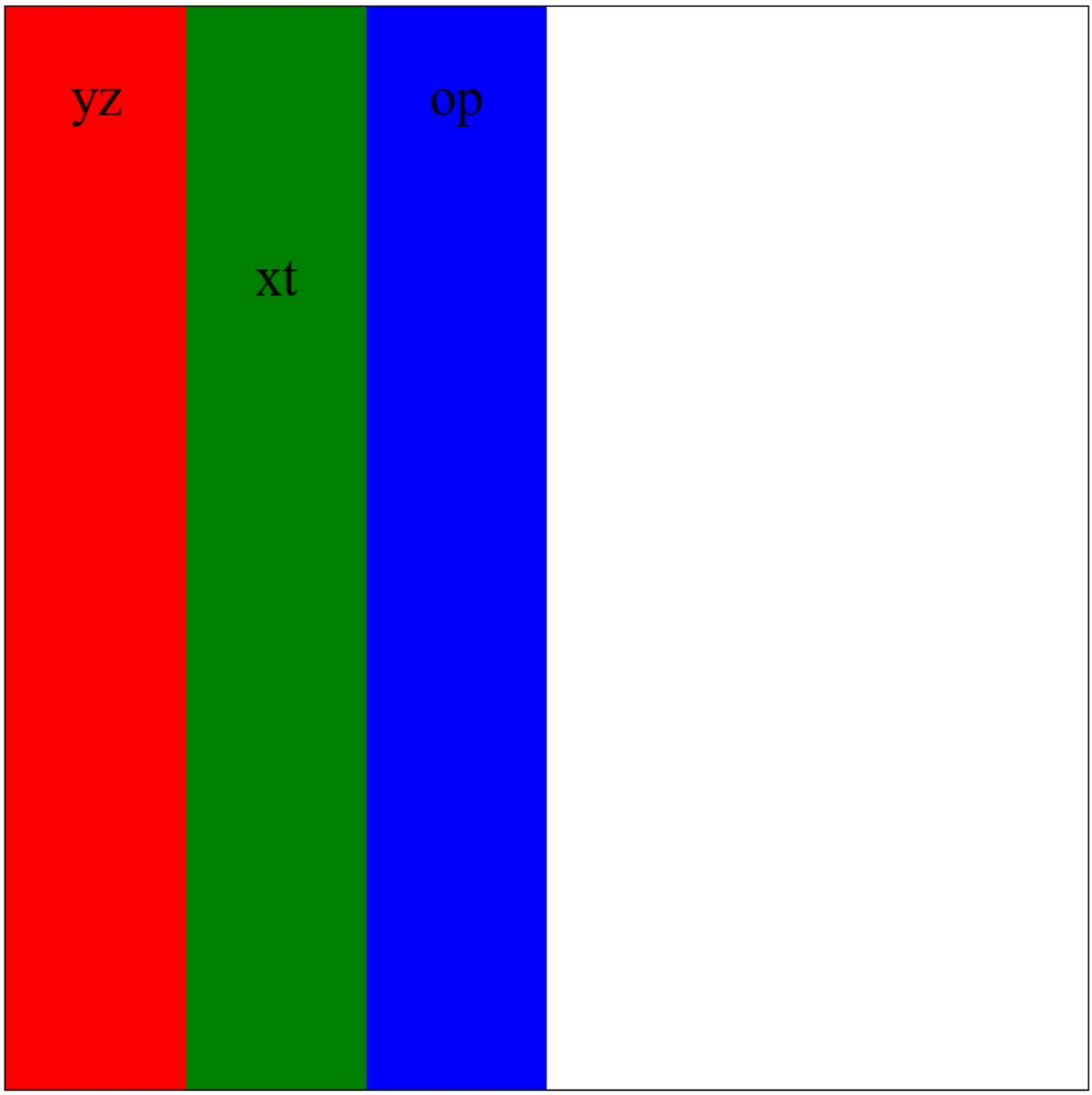
20. 伸缩布局-侧轴对齐方式(掌握)

基线即便没有最短文字，也在那里。

基线对齐图例：



拉伸对齐图例（伸缩项不能设置高度）




```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
```

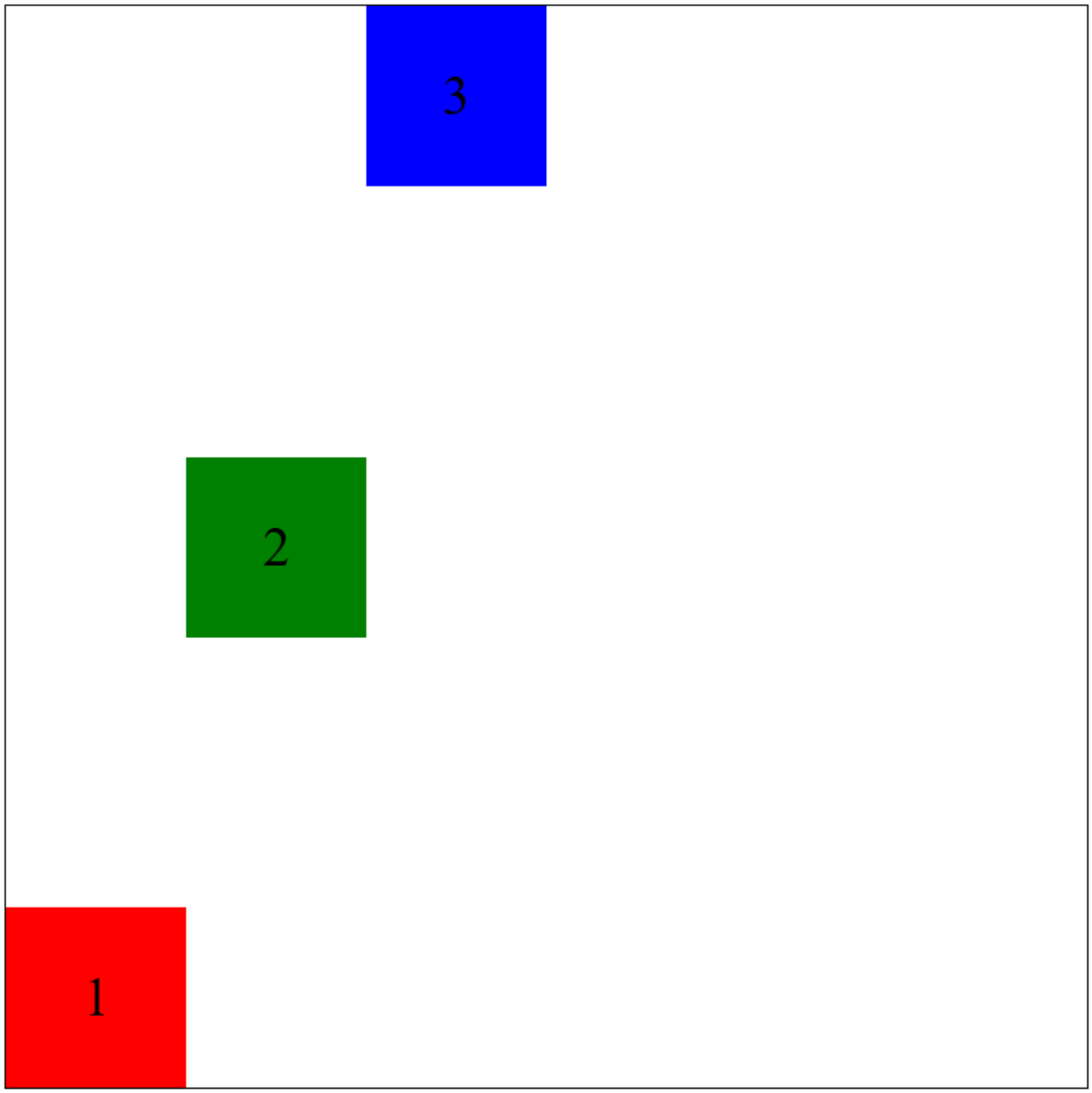
```
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
            /* 告诉浏览器，主轴起点在伸缩容器最左边，伸缩项从左至右 */
            flex-direction: row;
            /* 告诉浏览器排版好的伸缩项需要和主轴的起点对齐 */
            justify-content: flex-start;
            /* 告诉浏览器，排版好的伸缩项需要和侧轴的起点对齐，这就是默认值 */
            align-items: flex-start;
            align-items: flex-end;
            align-items: center;
            /* 注意点:
            侧轴对比主轴来说，没有两端对齐 space-between 和环绕对齐 space-around
            */

            /* 基线对齐 baseline:
            让所有伸缩项中的基线在一条直线上对齐 */
            align-items: baseline;

            /* 拉伸对齐/等高对齐
            让所有伸缩项的高度变为侧轴的高度
            注意点:
            如果需要设置为拉伸对齐，那么伸缩项不能设置高度
            如果伸缩项设置了高度，那么拉伸对齐就会失效。
             */
            align-items: stretch;
        }

        ul>li {
            width: 100px;
            /* height: 100px; */
            line-height: 100px;
            text-align: center;
            font-size: 30px;
            background: red;
        }

        ul>li:nth-child(2) {
            padding-top: 100px;
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>

<body>
    <ul>
        <li>yz</li>
        <li>xt</li>
```

```
        <li>op</li>
    </ul>
</body>

</html>
```

21. 伸缩布局-侧轴对齐方式2(掌握)

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;

        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
            background: red;
        }

        ul>li:nth-child(1) {
            /*
                如果在伸缩容器中通过 align-items 来控制伸缩项的对齐方式，
                是一次性控制所有伸缩项的对齐方式。
                如果想单独控制某一个伸缩项在侧轴上的对齐方式，
```

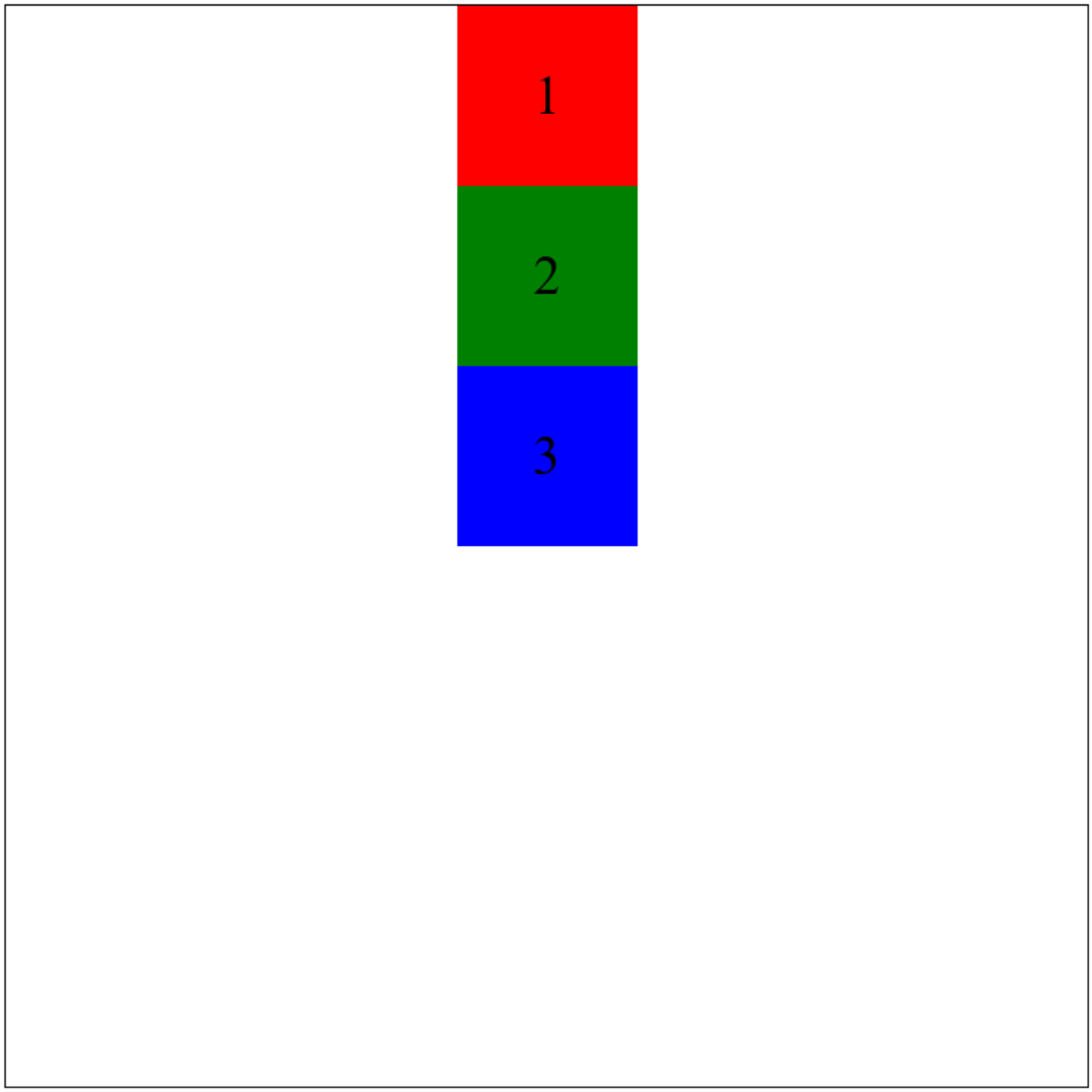
```
                需要将设置对齐方式的属性写到伸缩项中，
                而不是写在伸缩容器中。

                align-items：写在伸缩容器中，控制所有伸缩项。
                align-self：写在伸缩项中，控制编写对应代码的伸缩项。

                注意点：
                    align-self 的取值和 align-items 的取值是一样的，
                    只是控制的范围和书写的位置不同而已。
            */
            align-self: flex-end;

        }

        ul>li:nth-child(2) {
            background: green;
            align-self: center;
        }

        ul>li:nth-child(3) {
            background: blue;
            /* 这是默认情况 */
            align-self: start;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```



22. 主轴和侧轴交叉问题(掌握)

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }
```

```
        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
            flex-direction: column;
            align-items: center;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;
        }

        ul>li:nth-child(2) {
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```



23. 伸缩布局-换行和换行对齐(掌握)

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }
```

```
        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;

            /*
            1. 默认情况下，如果伸缩容器的一行放不下所有伸缩项，系统会自动等比压缩所有伸缩项，保证能够把它们放在一行。
            2. 在伸缩容器中有一个 flex-wrap 属性，专门用于控制放不下是否需要换行的。
                默认取值是 nowrap
                wrap：放不下就换行，而不是等比压缩
                wrap-reverse：放不下就换行，以行为单位进行反转
             */
            flex-wrap: nowrap;
            flex-wrap: wrap;
            flex-wrap: wrap-reverse;

            /*
            伸缩容器中，有一个叫做 align-content 的属性，专门用于设置换行之后的对齐方式。
            注意点：只有伸缩项发生了换行，这个属性才有效。

            align-content: flex-start;
            换行之后和侧轴的起点对齐，一行接一行。

            align-content: flex-end;
            换行之后和侧轴的终点对齐，将所有换行的内容当做一个整体来操作。

            align-content: center;
            换行之后和侧轴的中点对齐

            align-content: space-between;
            换行之后在侧轴上两端对齐

            align-conent: space-around;
            换行之后在侧轴上环绕对齐

            align-content: stretch;
            以行为单位进行拉伸，拉伸的部分以空白进行填充，保证拉伸之后所有的行加起来能够填满侧轴
             */
            flex-wrap: wrap;
            align-content: flex-start;
            align-content: flex-end;
            align-content: center;
            align-content: space-between;
            align-content: space-around;
```

```
            align-content: stretch;
        }

        ul>li {
            width: 300px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;
        }

        ul>li:nth-child(2) {
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

24. 伸缩布局-伸缩项排序(掌握)

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
```

```
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;
            /*
                默认情况下，每个伸缩项都有一个 order 属性，用于决定排序的先后顺序。
                默认情况下，所有伸缩项的 order 属性的取值都是 0，也就是说，谁写在前面，谁就排在前面。
                我们可以通过修改 order 属性的取值，来实现伸缩项的排序。
                排序规则：升序（从小到大的排序），越小的显示在越前面，越大的显示在越后面。
             */
            order: 999;
        }

        ul>li:nth-child(2) {
            background: green;
            order: 0;
        }

        ul>li:nth-child(3) {
            background: blue;
            order: -1;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
```

```
    </ul>
</body>

</html>
```

25. 伸缩布局-伸缩项扩充(掌握)

需求：想让每个伸缩项变宽，填满伸缩容器


```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }
```

```
        ul>li:nth-child(1) {
            background: red;
            /*
            在伸缩项中有一个 flex-grow 属性，
            用于控制当所有伸缩项的宽度总和小于伸缩容器宽度的时候如何扩充自己，以便于所有伸缩项宽度的总和能够填满整个伸缩容器
            默认情况下，flex-grow 取值是 0，表示我们设置的宽度是多少就按照多少来显示，不进行任何扩充
            注意点：
            只有当所有伸缩项的宽度总和小于伸缩容器宽度的时候 flex-grow 这个属性才有效

            flex-grow 扩充的公式
            1. 伸缩容器的宽度 - 所有伸缩项的宽度 = 剩余空间
                600 - 300 = 300
            2. 剩余空间 / 所有需要扩充份数的总和 = 每一份的大小
                300 / (1 + 4 + 8) = 23.07
            3. 当前伸缩项的宽度 + 需要的份数的宽度
                第一个伸缩项：100 + 1 * 23.07 = 123.07
                第二个伸缩项：100 + 4 * 23.07 = 192.28
                第三个伸缩项：100 + 8 * 23.07 = 284.56
             */
            flex-grow: 1;
        }

        ul>li:nth-child(2) {
            background: green;
            flex-grow: 4;
        }

        ul>li:nth-child(3) {
            background: blue;
            flex-grow: 8;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

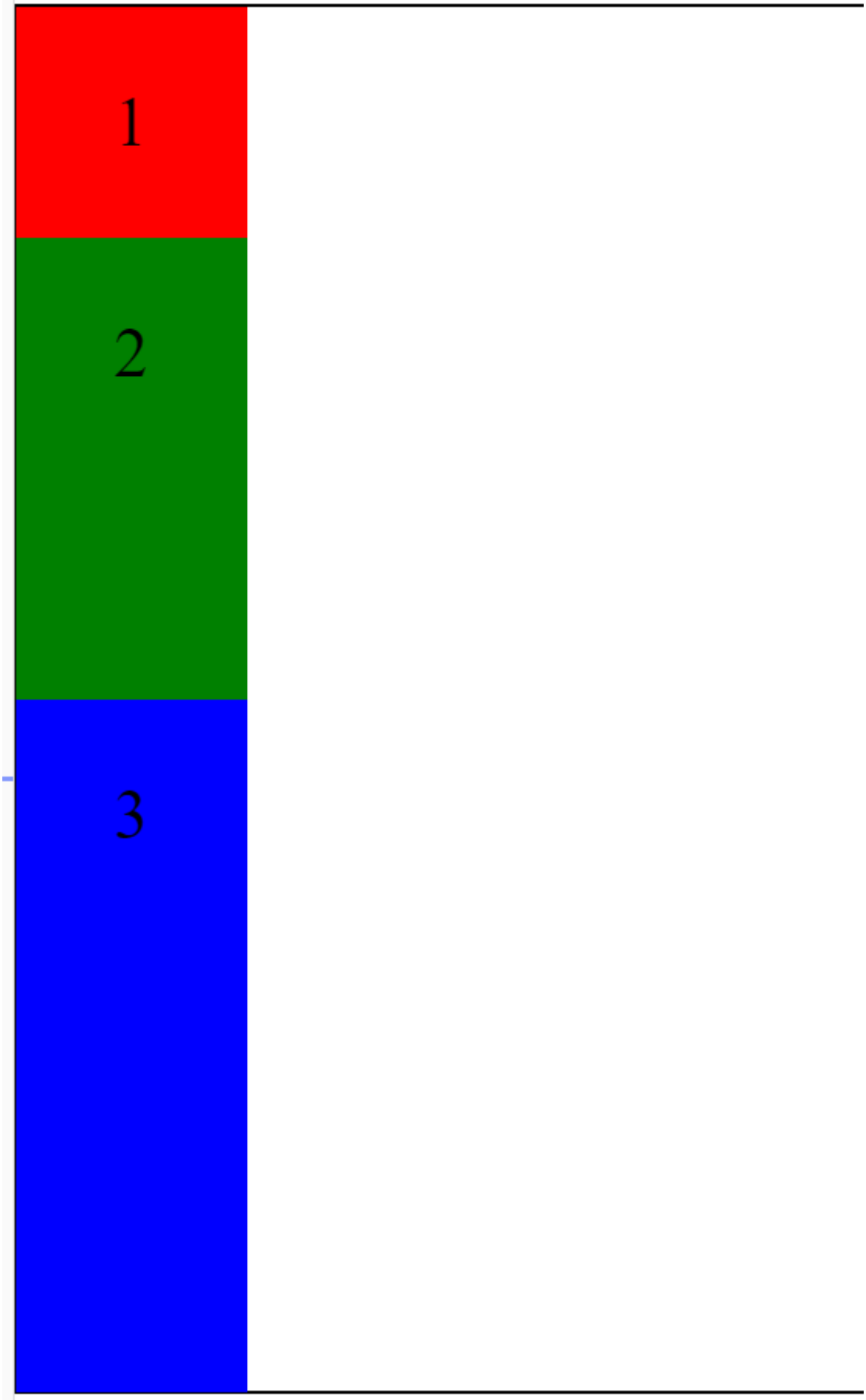
26. 伸缩布局-伸缩项缩小(掌握)

团子注：`flex-shrink` 值越大，则被压缩的越厉害

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
        }

        ul>li {
            width: 300px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;
            /* flex-shrink: 0; */
            flex-shrink: 1;

            /*
            在伸缩项有一个 flex-shrink 属性，用于控制当所有伸缩项的宽度总和大于伸缩
容器宽度的时候，
            如何缩小自己，以便所有的伸缩项宽度的总和可以填满整个容器。
            默认情况下 flex-shrink 的取值为 1，表示当所有伸缩项宽度的总和大于伸缩容
器宽度的时候，缩小自己

            注意点：
```

```
            只有当所有伸缩项的宽度总和大于伸缩容器宽度的时候 flex-shrink 这个属性才有效。
            flex-shrink 取值为 0 的时候，则不会压缩伸缩项，而是会超出伸缩容器的范围。

            flex-shrink 缩小的公式
            1. 所有伸缩项的宽度总和 - 伸缩容器宽度 = 溢出的宽度
            900 - 600 = 300
            2. 计算权重值
            每一个伸缩项需要的份数 * 当前伸缩项的宽度，然后再相加
            1 * 300 + 4 * 300 + 8 * 300 = 3900
            3. 计算每个伸缩项需要缩小的范围
            溢出的宽度 * 当前伸缩项的宽度 * 当前伸缩项需要的份数 / 权重值
            300 * 300 * 1 / 3900 = 23.07
            第一个伸缩项的宽度 = 300 - 23.07 = 276.9
            300 * 300 * 4 / 3900 = 92.3
            第二个伸缩项的宽度 = 300 - 92.3 = 207.6
             */
        }

        ul>li:nth-child(2) {
            background: green;
            /* flex-shrink: 0; */
            flex-shrink: 4;
        }

        ul>li:nth-child(3) {
            background: blue;
            /* flex-shrink: 0; */
            flex-shrink: 8;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

27. 伸缩布局-扩充缩小注意点(掌握)




```
<!DOCTYPE html>
<html lang="en">

<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
            flex-direction: column;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;
            /*
            1. 如果没有指定 flex-grow 属性，或者 flex-grow 的值为 0，那么当前的
伸缩项不会被扩充。
             */
            flex-grow: 0;

            /*
            2. 如果 flex-shrink 的值为 0，那么当前的伸缩项不会被缩小
            3. 注意点：
            前面所写的注释都是宽度扩充或者宽度缩小，但是这种说法是不严谨的
            也有可能扩充和缩小的是高度，到底是宽度还是高度是由主轴决定的
            扩充和缩小的是主轴方向上的值
            也就是说，如果主轴是水平方向的，那么扩充和缩小的就是宽度
            也就是说，如果主轴是垂直方向的，那么扩充和缩小的就是高度
             */
            /* flex-shrink: 0; */
        }

        ul>li:nth-child(2) {
            background: green;
            flex-grow: 4;
```

```
            /* flex-shrink: 4; */
        }

        ul>li:nth-child(3) {
            background: blue;
            flex-grow: 8;
            /* flex-shrink: 8; */
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

28. 伸缩布局-宽度设置(掌握)

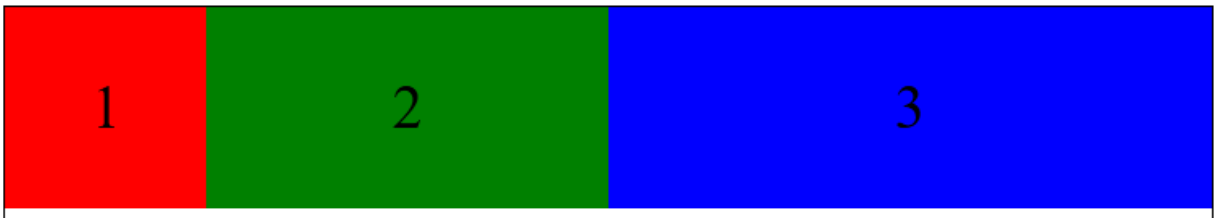
伸缩项如果不设置宽度，那么它的宽度将由其内容撑起。
`li` 标签是块级元素，默认和父级一样宽。
`width: auto;` 内容宽度撑起元素宽度。

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
```

```
            margin: 100px auto;
            display: flex;
        }

        ul>li {
            /* width: 100px; */
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;
            /*
                1. 在伸缩布局中可以通过 flex-basis 属性设置伸缩项的宽度
                注意点：flex-basis 只有在伸缩布局中才有效
                2. 在伸缩布局中，如果通过 flex-basis 设置了宽度，那么再通过 width 设置宽度就会无效。
                也就是说，flex-basis 的优先级要高于 width 的优先级
                经测试，连 !important 也无法覆盖。
                3. 在伸缩布局中，如果同时通过 flex-basis 和 width 设置了宽度，
                而且一个设置的是 auto，另一个设置的是具体的值，那么会按照具体的值来显示，
                并且与前后顺序无关。
            */
            /* flex-basis: 100px; */

            /* width: 200px; */
            /* width: auto; */

            /* width: 200px; */
            /* flex-basis: auto; */

            width: auto;
            flex-basis: 300px;

            flex-basis: 300px;
            width: auto;
        }

        ul>li:nth-child(2) {
            background: green;
        }

        ul>li:nth-child(3) {
            background: blue;
        }
    </style>
</head>

<body>
```

```
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

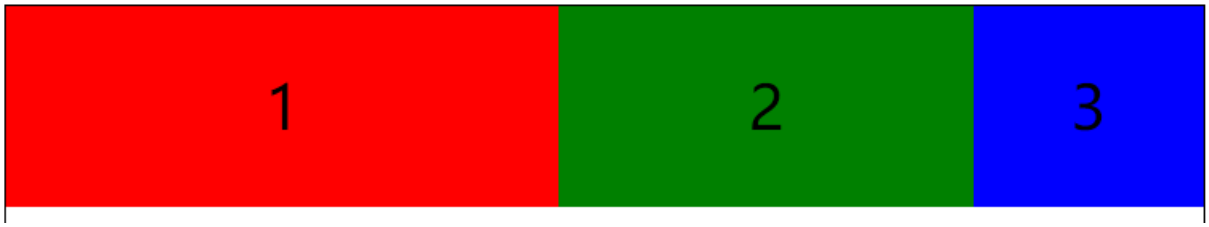
29. 伸缩布局-伸缩项属性连写(掌握)




```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
```

```
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;

            /*
            flex: 扩充 缩小 宽度;
            */
            /* 下面是默认值，所以不会有任何变化 */
            flex: 0 1 auto;

            flex: 0 1 100px;
        }

        ul>li:nth-child(2) {
            background: green;
            flex: 0 1 200px;
        }

        ul>li:nth-child(3) {
            background: blue;
            flex: 0 1 300px;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

29. 伸缩布局-伸缩项属性连写(掌握)




```
<!DOCTYPE html>
```

```
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>伸缩布局</title>
    <style>
        * {
            margin: 0;
            padding: 0;
        }

        ul {
            list-style: none;
            width: 600px;
            height: 600px;
            border: 1px solid #000;
            margin: 100px auto;
            display: flex;
        }

        ul>li {
            width: 100px;
            height: 100px;
            line-height: 100px;
            text-align: center;
            font-size: 30px;
        }

        ul>li:nth-child(1) {
            background: red;

            /*
            flex: 扩充 缩小 宽度;

            flex 默认值:
            flex: 0 1 auto;
            */
            /* 下面是默认值，所以不会有任何变化 */
            flex: 0 1 auto;

            flex: 0 1 100px;
        }

        ul>li:nth-child(2) {
            background: green;
            flex: 0 1 200px;
        }

        ul>li:nth-child(3) {
```

```
            background: blue;
            flex: 0 1 300px;
        }
    </style>
</head>

<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
    </ul>
</body>

</html>
```

完成于 `2019.8.16`